CHAPTER 8
 Measurement & Results analysis

8.1 Testing – Clustering and Failover

Today we have lots of application server application exist in the market but oracle weblogic server is one of the best application server having much more capabilities in comparison with others to make you applications highly available and scalable. Specifically in terms of scalability you can add more and more managed servers at run time to get more scalability and the server migration and clustering feature make the applications highly available. Only the limitation that it does not provide you is a single shared ip address to handle outside world http requests.

In cluster it’s not recommended or allowed to connect you with a specific managed server hosting the application. So for these capabilities oracle provided a plug-in for the web servers which listen on a single address for the outer world http requests and forward requests to the multiple clustered managed servers in round robin basis. But also lead to a single point of failure at web server ip address.

Here we need some additional software to monitor for example to monitor the web server processes, and in case of failure of the web server processes this software should failover the i.p address to the another standby server so that web server should work properly. This can be possible but I will test here with the open source Linux.

Required Software’s

Oracle Enterprises Linux 5.4 ( with OEL, Development, Cluster etc ), with this you will get complete tools and libraries to compile the needed applications and software’s. Apache web server also included with this pack.
Results

During my testing I found that the most efficient algorithm for the load balancing was round robin and this was specifically to my testing environment where each hardware were of same configuration across the all managed servers participating in the clustered way.

Another tested algorithm was weight based. Its useful if you have hardware with the different configurations. Like suppose if you have machine A with 2 cpu and 2 GB RAM and machine B with 1 cpu ang 1 GB RAM then definitely you want more traffic on machine A with respect to the traffic on Machine B. So I have changed the weitage percentage on server D to 100% and 50% on server DCL2. During testing the results was as expected, 1/3 load on second server and 2/3 on first server.

Last one tested algorithm was random, test results shown that the random algo results were almost the same as the results of the round robin. This was may be due to I had only two options in the cluster or the tests was being uniform and program was able to predict[34].

Cluster Algorithm graph

Fig 20
the weblogic inbuilt mechanism to logging the requests to each and every server and configured the same.

Another challenge was, since weblogic server is the J2EE based application server, to build the java based test program in J2EE format. I found that a normal java program is different than writing the program in J2EE standard. So studied further and later I was able to create and deploy a testing java application.

Another challenge which I didn’t realized initially was I have configured lots of logic in the descriptors of the deployment. It took a lot to me to get only used one descriptors and to remove not required one to get it functioning properly.

Rest of the issues I found during the testing specially in the testing of failover capabilities. I have done complete setup according to the requirement and the standard document but wasn’t being able to get the results. Failover has not been worked first for the cache and beans level on the cache. Initially I have a very low value for the cache and found lots of exceptions on the first server only not on second server. Failover was not utilizing the objects properly, later I have killed one of the server to check if requests diverting to other server or not.
CHAPTER 7

Test Setup

To understand the concept of failover and load balancing in weblogic, I ran multiple tests to gather the data and to determine the efficiency and effectiveness of all the three available algorithms. In my LAB environment, I have created a domain with three weblogic server instances on machine A and two more clustered servers on another machine B. The clustered servers were named as D and DCL2.

For testing I have created a stateful session bean and deployed it on the created clustered. This bean only contained a ping method which on return simply prints a string. Client java program will create lots of threads and will invoke the deployed session bean many times in a row and will make an entry in log file for each receiving request. Since the request will divert in cluster so it will be logged in the different server logs according to the algorithm. Later I have examined the logs to know how the loads or requests were distributed[32][33].

Challenges Faced

Throughout the testing of this concept I got lots of challenges and problems and out of all on some I was able to overcome however on some not. The first and the main challenge was the setup of the domain and cluster on the weblogic about which the information is not very clear anywhere, also the documentation is not so good enough and helpful to determine the best settings like beans in the cache. This caused lots of testing to setup the domain, cluster and deployment to achieve the desired solution.

Another challenge was how to configure the logging and collect the statistics of the tests. Logging is the main method to capture and troubleshoot the real picture of processed requests. Since my concern was not all the requests goes to a single server instead of that equally distributed over the cluster so it was the main work me to find out the proper solution. I tried to used the java options like print line but it doesn’t work for me properly. After doing further investigation I found
For EJB’s and RMI objects load balancing and failover is achieved using the replica aware stubs, which may locate the instance of the object within the cluster. Sometime we called it smart stubs. These replica aware stubs are created and handled for RMI and EJB’s as a result of outcome of the compilation process of objects. So all the RMI and EJBs are homogeneously deployed across all of the servers in the cluster.

Weblogic supports different kind of cluster algorithms for load balancing the EJB’s and RMI’s like round robin, weight base and ransom. Whichever the method you will select will be maintained within the replica aware stub used for the objects clustered.

**Server Communication**

For proper load balancing, servers within a cluster communicate with each other using the multicast, unicast or IP sockets.

Servers within the cluster communicate with each other using any one of the above communication technology. Using the IP multicast utility servers used the simple mechanism of broadcast in which each and every server communicate with each other using the heartbeat mechanism where they send messages to each other to let each know the availability of the other system. A multicast I.P defined in a given selected defined range from 224.0.0.0 to 239.255.255.255 and port cab be used anyone which is not in used. Using multicast which is one too many communication each and every server communicate with each other in the cluster.

- **JNDI updates-Cluster-wide** –

  Every server in the cluster communicate with each other server to inform about his availability using the multicast address and port, each server then listen for the updates from other server and update its local JNDI tree to update the running deployment of the clustered objects.

- **Heartbeats in Cluster**–

  every server in a cluster use multicast to send heartbeat to other servers to inform then its availability, till all the servers listen each other be in synch in cluster and in case of any server goes down cluster remove that server from the cluster.

- **IP sockets are used for the peer to peer communication between the servers and used when two servers in the cluster communicate with each other.**
Another great feature of weblogic server clustering is load balancing which allow the equally distribution of the jobs or as per defined cluster algorithms among the servers within a cluster.

To achieve load balancing:

- There should be more than one copy of an object that perform the job
- For each and every object the operational and location status should be available

Weblogic allows all of the objects are clustered and deployed across all of the clustered managed servers so that in case of failure of an object other one alternative object will perform the rest of the job.

**Type of Clustering in WebLogic Server**

We have lots of resources in the field of information technology and weblogic supports clustering of various kind of resources. Each and every resource having a different kind of behaviors like the invocation, related to control and functioning with in an application. So the clustering benefits of load balancing and failover varies for different kind of resources and weblogic provides support for different kind of resources[31][32].

Below are the objects clustered in weblogic server

- Servlets
- JSPs
- EJBs
- Remote Method Invocation objects
- Java Messaging Service destinations
- Java Database Connectivity connections

**How It Works**
BEA WebLogic Clustering

The two great features of the weblogic server make it to achieve the high availability and scalability and these capabilities are the load balancing and failover. Failover means suppose if a connection is get connected with a particular session on a particular server and suddenly that server goes down then the complete session of that connection will failover to the other server in the cluster.

So to meet the condition where newly created objects should take over the failed objects below mentioned should need to be met -

- There should be a copy of each session should be maintained
- To know the exact behavior of a object if it is failed, the information of each object should available to other failover object and also to program which handle the failover like location the status of all of the objects.
- All the information about the currently running jobs should make available to all the objects and programs manages the failover. So in that way an object which take over the failed objects know about where it was during failure, what was that doing and what need to process next.

Weblogic use J2EE based standard technologies and some internal inbuilt technologies like JNDI (java naming and directory interface), multicast, unicast, IP socket etc to maintain and share the information about the availability of the objects in the cluster. All these features and technologies allow weblogic to determine that the particular functionality has been interrupted and the stage where it got failed and the backup failover server where this needs to be replicate.

All the current running jobs information are stored in a logical object to which we called state and weblogic maintains the details of each and every state using the inbuild technology called replica aware stubs and session replication. In case of failure of a particular session stub and session replication allow that job to failover to the other live objects from the same place where it was failed.
CHAPTER 6

Clustering Requirements and Types

Clustering is one of the best high availability solution in the IT industry to achieve proper load balancing, failover and session replications or you can it’s a perfect business continuity solution nowadays. It’s totally transparent to the end users accessing the applications and cluster act as a single service instance for the end user.

Oracle Weblogic server is the best clustering supported application server in the market today having the good high availability feature solutions. Below testing has been done on the same and I will describe the different algorithms support with the weblogic for proper load balancing.

Clustering Requirements

For better solution, Clustering should support the following requirements -

First, Applications must continue to serve the requests of clients without interrupting the new and current connections and for that there shouldn’t be and bottlenecks for the further scaling. If needed we can add more application servers to the cluster at run time to met the increased demand of the users so in that way you should be able to do the dynamic changes at the run time.

Next, designed architecture should design to maintain high availability. If any server goes down or fail in the cluster the requests should not impacted and continue to processing the requests in the cluster. Application should deploy homogeneously on all the servers in a particular cluster. If any one of the server went down then the replica of same should be failover to the other server in the cluster. Applications should not be connected with the cluster features and developers should not have any dependency on the cluster features like load balancing, failover and replication so that applications can design independently and can be deployed as platform independent anywhere.

And the final requirement is that, the clustered services should be managed in way as the non implemented clusters. A cluster should appear as a single point to the system administrator and administrator should able to handle a cluster a single server[31].