International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

An Approach for Horizontal Scaling of Cluster

Manodaya B. Gavali[1], Navnath Kale[2]

[1, 2]University of Pune, PVPIT College of Engineering, Bavdhan, Pune-21, India

Abstract: ***Cloud computing is nothing but the delivery of computational power in the form of service and not in the form of a product, here shared resources, software and data are given to computers and other devices as a metered service over a network. Platform as a Service is a one of the cloud based approach that provides companies all the functionality for developing, deploying and managing the services, without any load to configure, install and manage the middle ware, hardware and operating system. The objective of this framework is "To provide an event driven framework to manage computing platform based on demand". System is proposed to organize middleware and virtual machine to provide High Availability clusters. A deployment diagram is used to catch the idea of topology of a cluster used in cluster scaling. This framework provide facility to scale out/in resources in cluster, when load at the cluster is increased then new node is created and attached to cluster according to configuration given in deployment diagram, also when load decreases resources are released to free pool.***

Keywords: cloud, IaaS,PaaS,SaaS, HA cluster, Scalability

1. Introduction

Cloud computing is a model for enabling, presenting a suitable, on-demand network access to a shared pool of configurable computing resources such as network, storage, server in form of compute, applications complying SOA, and frequently provision and release of services with minimum management effort or minimum interaction of service provider. Applications deployed on cloud computing environment have shown plenty built-in advantages. Flexibility and High Availability is one of the important advantages of them. For example, the data and files can be acquired by Staff in Storage cloud as per there requirement even though they are working remotely and outside office. Consider, for example, how quickly a cloud enabled application can scale up/out as and when needed. Cloud computing is less costly and without any manual intervention. Instead of huge investment at the starting phases of project, cloud computing aims on sharing the resources, software is provided ”on-rent” basis.

2. Problem Statement

Load on the system is unpredictable. To manage that load we need to manage cluster size which requires manual intervention. This can slow down the system performance. It can be overcome with the automated scale in/out. In this scope of the paper we are trying to solve three problems:

1) Automation of cluster installation with the use of deployment diagrams and related scripts which will be installing and configuring the middleware.
2) Monitoring the load on the system; after installation of the cluster, the middle ware management scripts should be capable to utilize further to monitor the load on each of the middleware.
3) Depending on the load on each node of all the middleware, if the load crosses a predefined threshold, automatically a node should be provisioned, middleware should be installed and configured and plugged into the cluster. And if the load crosses the below threshold, we unplug the node and destroy it.

3. Literature Survey

Cloud computing provide Scalability as one of the key benefits. Ability of system to enhance its capability to manage larger loads by put in the nodes or resources at runtime is called Scalability. It is possible to easily upscale or down scale the cluster on the basis of business requirements. Consider the example of banking applications which is are heavily used in day time and not used in evening and weekends, in such relaxed time the servers can be used by other time zone organizations, and it can be used by travel organizations on holidays. Existing resources can be increased by service provider according to changes business needs without any need of costly changes existing IT systems.

There are two types of scaling: Horizontal and Vertical scaling [3]. Horizontal scaling is used when it is impossible to change one resource type to other type. If it is not possible for IaaS layer to scale up virtual machine at runtime because of host machine resource processing or hypervisor inability, horizontal scaling is chosen by PaaS via scale out and scale in such cases. In this nodes are scale out or scale in to the system as per needs with less RAM and processor. In vertical scaling we add (i.e. scale up) or remove (i.e. scale down) resources such as processor and RAM in the system. Horizontal scaling is easy to implement but quite costly than vertical scaling. Applications needs middleware support like web servers, database, mail server, application servers, proxy server, file server, message queuing system etc in modern day. We need to manage runtime deployment of middleware, initializing middleware and adding the server to the cluster application used to deploy and scale down, up the application at runtime. Shortfall of such type of framework tends to the failure of applications to get deployed over cloud environment. Similar kind of work proposed Hailong Sun[1] for Service Oriented Architecture (SOA) used for deploying SOA based solutions on PaaS . This system contains merging of the Deployment model and scaling in/ out of platform for generic cluster applications.

One more work is proposed by Bao Rong [2] for enterprise resource planning to provide high availability and scalability

to it. The WebSphere cloudburst [8] is solution provided by the IBM for resource monitoring, designed to speed creation and deployment of application to cloud and virtual environment. This earlier solution does not consider the load on the individual server into account to scaling out/scaling in the cluster. Proposed framework has the facility so that virtual nodes in the cluster can send information about the health and load the system is going through which can be used for automated deployment.

High availability cluster is group of computers working to minimize chance of complete downtime. It avoids single point of failure, if one component is failed then its services taken over by other redundant component until crashed server start working. HA cluster is used by critical db, file sharing system and web app. Many nodes can perform same role and work together in HA cluster. Node configuration is categories into different models such as Active/active, Passive/active, N+1.Set of UML representation named as - deployment diagrams used to represent HA architectures; which is used for physical entities deployment on the node. It composed of node and their relationship with each other and used to narrate the systems static deployment view and hardware component on which software components deployed. Performance, maintainability, scalability and portability can be controlled by efficient deployment diagram. HA architecture can be captured with the UML diagrams along topology files.

4. Proposed Framework

Proposed system deals with PaaS service model and deployment diagrams used to capture the HA Cluster configuration and aims to satisfy all the four essential characteristics of cloud. This system provide framework for on-demand and automated scaling with event driven mechanism. Virtual node on the each cluster report back to the monitoring server regarding its health and load. Component of particular role is then scale out when load at the virtual node increases and the agent present at each node report back to the monitoring server about increased load, then. Similarly, that particular role component is then scale in if load at virtual node decreases and the agent present at each node report back to the monitoring server about decreased load. Deployment diagram and scripts are given to management server for configuration.

PaaS Management Fabric (PMF) is a framework to manage virtual machine and middleware including its health as well as underlying virtual machine health. Architecture consists of an agent running on all the machines of a cluster and another server from where all the operations are driven. Each platform will be defined by a topology file, which provides information about middleware; its configuration parameters and code deploy scripts. This topology along with deployment scripts are provided by user. PMF will read the topology file and deploy middleware followed by applications deployment. These scripts and installers will be kept in File Repository.

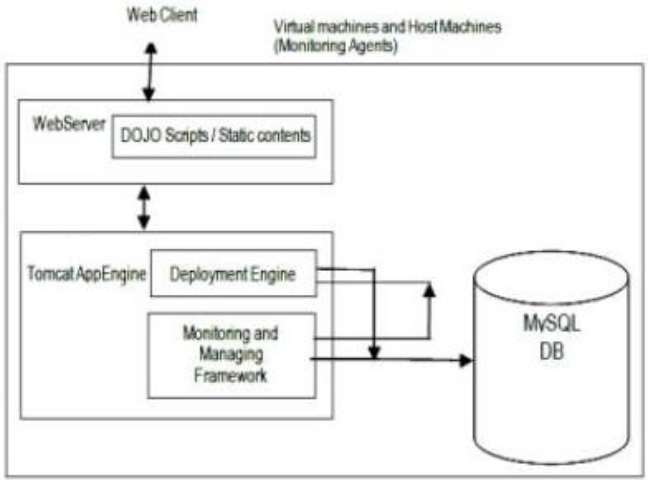

Figure 1: System architecture of proposed system As shown in above figure the proposed system works as follows:

User who wishes to use this framework need to provide deployment details and configuration to the system admin of PMF. Which then create deployment diagram and topology files from provided details. Meanwhile agent present at each node sent back report about its health regarding load on it to monitoring agent. Which is then report to the workflow execution engine about load on particular role of cluster, if load crosses the threshold value given in topology file then new node of that role added by using deployment diagram else if load is less than threshold value then unplug the node of that role.

The system contains the following modules:

1) Workflow Execution Engine: Workflow Execution Engine will be responsible for vm images installation, provide network connectivity to them, middleware installation and configuration. Workflow execution engine from the PMF has cleaned the database environment and started itself waiting for user to initiate further operation. When the deployment request of particular roles comes from the Monitoring Broker when load at the cluster for then workflow execution engine will responsible for deploying that role. As shown in the right side of the figure, we can see the virtual machine has been deployed in VirtualBox.
2) Monitoring Broker: Monitoring Broker will read script file and create definition of a base platform, facility to modify this definition through WEB User Interface. Following information available in Platform definition.
   a) Platform architecture including multiple middleware and specifications of VM.
   b) Rules for deployment:

Platform architecture defines the architecture required by application for run and Rules; for deciding when to deploy an application and destroy it. Monitoring Broker will take help from plug-in related to different middleware to deploy, manage and destroy them. This module uses a MySQL as database and will provide RESTful interface for web client. After installing and configuring whole cluster now the PMF has entered in monitoring mode. In this mode the PMF will continuously monitor all the machines of the cluster one by

one of each role, in roles sorted in topological order. PMF will wait for 1 minute between each run of monitoring.

3) Web User Interface: Web User Interface will provide overall management of framework, consist of:
   a) Insertion and modification of topology file.
   b) Topology files testing, by checking whether it will deploy successfully or not.
   c) Installation of middleware.
   d) Report back platform health.

Web User Interface is a scripts running on RESTful framework.

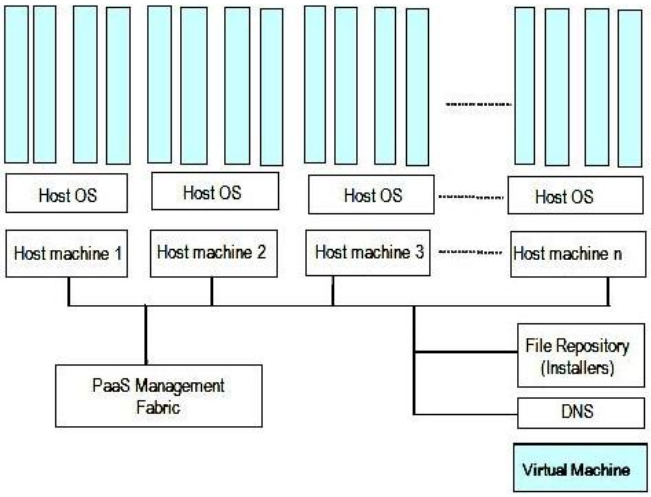

Figure 2: System architecture of proposed system

Above figure shows architecture of system which uses PMF. Monitoring broker from PMF monitor system for load on it. If load increases then Workflow execution engine create the node by using deployment script present in the file repository for particular role. Then IP address assigned to it using the DNS.

The load on the cluster might be compensated by adding a round of servers in the cluster. After adding machines, there might be case that there could be the case that the clusters are still under stress. This causes the PMF to not to be sufficiently responsive towards the variable load. PMF should be checking the earlier history and find out the pattern in which load is changing over the cluster. To make the system more adaptable to newer workload, PMF need to be changed to learn from the earlier pattern and predict the future load by regression model and add that many number of machines before hand.

In this paper we have used linear regression to predict the number of machines the cluster would require in near future. To avoid contamination of the earlier trends, the linear regression model takes into account latest 4 results to predict how many nodes are required in future.

5. Mathematical Model

Let system S=(h, v, r)
  Where;
  h is hypervisor such that h=h1, h2, h3..
  v is set of cluster VMs v=v1, v2, v3,..
  r is role r=r1, r2, r3,..
while true
  Test the load on each cluster vm of each role
  If more number of vm are under stress than threshold then add the cluster vm of that role
    Download the script and image mount image
    Configure the IP address and copy role script unmount image
    Deploy the image configure the role plug-in vm to cluster
  If less number of node are under stress than the threshold and there are more number of vm allocated to the role than initial configuration
  Then
    randomly select and delete cluster vm unplug the vm from cluster unconfigure the role
    destroy the image
    delete image file
  sleep for one minute repeat the procedure

As in mathematical model first step contains the set of system which contains set of hypervisor (h), set of cluster virtual machine (vm), set of role (r). After system setup is done, Monitoring broker will test load on each vm in cluster. Agent presents at each vm report back about its load to

Monitoring agent. If the less number of vm than the threshold value is under stress then randomly select the cluster vm and delete it by unplugging the cluster from vm, unconfigure the r

ole, destroy the image and delete image file. Otherwise if more number of vm under stress than threshold value then create the vm of particular role using role script by downloading script

and image, mount the image, configure IP address for that vm , deploy the image, configure that role and plug-in it into cluster. This process will repeat continuously to check load on cluster.

6. Algorithm for Addition of Cluster Node

addCluster(role,
isNewDeploymentRole=False): IPAdd =
        getNextIPAddress() host = getHost(role) scriptPath =
composeRoleDeploymentFile(role,host,IPAdd)
        copy topology file to host machine execution of
        copied script from
host machine
        until (virtualMachine is started)

        sleep(1);
        coping and executing script file for role
        configuration to virtualMachine.
        if isNewDeploymentRole:
        for roleDependency depending on
role
                for vm of roleDependency Notification to
                vm about
addition of new node

```
    else:
            for all roles dependent on the role 'r'
            for vm of roleDependency Notification to
            vm about
addition of new node
```

7. Algorithm Used for Removing the Node From Cluster

```
RemoveCluster(node):
        for roleDependency dependent in node.RoleName:
           for vm of role roleDependency.RoleName:
              Notification to vm about removal of node
        unconfigure vm
        stop middleware on that vm
```

8. Linear Regression Model to Predict Future Load

The load on the cluster might be compensated by adding a round of servers in the cluster. After adding machines, there might be case that there could be the case that the clusters are still under stress. This causes the monitoring broker to not to be sufficiently responsive towards the variable load. Monitoring broker should be checking the earlier history and find out the pattern in which load is changing over the cluster. To make the system more adaptable to newer workload, monitoring broker need to be changed to learn from the earlier pattern and predict the future load by regression model and add that many number of machines before hand. In this paper we have used linear regression to predict the number of machines the cluster would require in near future. To avoid contamination of the earlier trends, the linear regression model takes into account latest 4 results to predict how many nodes are required in future.

Linear Regression Model Algorithm

Q[4] = Queue of last 4 results of how many servers were required.

P = Future value

Here we need to find out the future value P, by considering past values of array Q.

here x is subscript of Q[] y is value for Q[]

Calculate:

a = SUM(y).SUM(x^2)-SUM(x).SUM(xy) / n.SUM(x^2) - SUM(x)^2

b = n.SUM(xy)-SUM(x).SUM(y) / n.SUM(x^2) - SUM(x)^2

P = a + b*5

9. Result of Implementation

On the increase of load, the nodes are added to the cluster without any intervention, and same when reduced, the nodes were swiftly removed from the system. While doing the testing this architecture shown high degree of sensitivity to the load and quickly acted on addition/removal of the nodes thought all the middle wares of the cluster. The resultant system can handle load increase upto 1 node per minute per middle ware. This time can be reduced by working on the details given in future scope. The time complexity required to increase the load is O(n); linear. Time complexity while reducing the load is O(1); constant.

10. Future Work

In future the proposed framework could be enhanced on following dimensions:

1) Automatically creating provisioned image of a middleware which could be just cloned and used for further deployment.
2) Delayed destroying of a node, if the load quickly increases after un-provisioning the node, we can utilize the already available node.
3) Keep pool of running nodes similar to thread pool, which can be used when case of urgency.

References

[1] Hailong Sun, Xu Wang, Minzhi Yan, Yu Tang, Xudong Liu "Towards a Scalable PaaS for Service Oriented Software", 2013 IEEE International Conference on Parallel and Distributed Systems
[2] Bao Rong Chang, Hsiu-Fen Tsai, Ju-Chun Cheng, Yun-Che Tsai High Availability and High Scalability to in-Cloud Enterprise Resource Plan- ning System", Intelligent Data analysis and its Applications, Volume II Advances in Intelligent Systems and Computing Volume 298, 2014.
[3] Bin Claudio A. Ardagna, Ernesto Damiani, Fulvio Frati, Davide Rebec- cani, "Scalability Patterns for Platform-as-a-Service", 2012 IEEE Fifth International Conference on Cloud Computing.
[4] X. Suo, Y. Zhu, and G. S. Owen, "CLOUD RESOURCE PROVISIONING AND BURSTING APPROACHES", 2013 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing.
[5] ZENG Shu-Qing "The Improvement of PaaS Platform" 2010 First International Conference on Networking and Distributed Computing
[6] Srijith K. Nair, Sakshi Porwal, Theo Dimitrakos, Ana Juan Ferrer, Johan Tordsson, Tabassum Sharif, Craig Sheridan, Muttukrishnan Rajarajan and Afnan Ullah Khan., Towards Secure Cloud Bursting, Brokerage and Aggregation, 2010 Eighth IEEE European Conference on Web Services
[7] NishantAgnihotri1,AmanKumarSharma EVALUATING PAAS SCAL- ABILITY AND IMPROVING PERFORMANCE USING SCALABIL- ITY IMPROVEMENT SYSTEMS, IJRET: International Journal of Re- search in Engineering and Technology eISSN: 2319-1163
[8] http://www-lia.deis.unibo.it/Courses/RetiM/RetiM-0910/seminari/IBM WebSphere CloudBurst.pdf
[9] Amazon Elastic Compute Cloud (EC2), http://www.amazon.com/ec2/
[10] The NIST Definition of Cloud computinghttp://csrc.nist.gov/publications/nistpubs/800-145/SP800.pdf
[11] http://webspherecommunity.blogspot.in/2010/01/websphere-cloudburst-appliance.html
[12] Google App Engine, http://appengine.google.com