ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Efficient Model Based Load Balance on Cloud Partitioning for the Public Cloud

Tellabati Nagaraju
M.Tech Student,
Department of CSE,
Pace Institute of Technology and Sciences, Ongole.

S.Phani Kumar
Asst Professor,
Department of CSE,
Pace Institute of Technology and Sciences, Ongole.

B.Srikanth
Asst Professor,
Department of CSE,
Pace Institute of Technology and Sciences, Ongole.

Jagadeeswara Rao.Annam
Associate Professor & HOD
Department of CSE,
Pace Institute of Technology and Sciences, Ongole.

Abstract:

Technology has its own implementation towards the automation system of Humanity considering all the aspects. In the recent trend of Cloud bomb where Software Giants want to do some changes towards next phase of Technology revolution takes us to next level of technology with all loop holes based software technology.

In the paper, after considering all the modes of cloud computing we keep forward the partition based parallel distribution system to be applied in the cloud environment where security in the public cloud would be the big issue. The Encryption and methodology involved in the into provide the best of service to mankind of the best in order to balance the load, which ni9n tem We named as Model Based ; which depends upon the peak and other factors of load .

Key words:
load balancing model; public cloud; cloud partition; game theory.

1. Introduction:

As cloud computing is growing in popularity it is also growing in complexity. More and more providers are entering the market and different types of solutions are made. There are few physical restrictions on how a provider should let their users do provisioning, and little limitations in technological solutions. The result can be a complex and struggling introduction to cloud computing for users, and provisioning procedure can alternate between providers. First the scenario will be introduced, describing the example application and different means of provisioning in form of topologies.

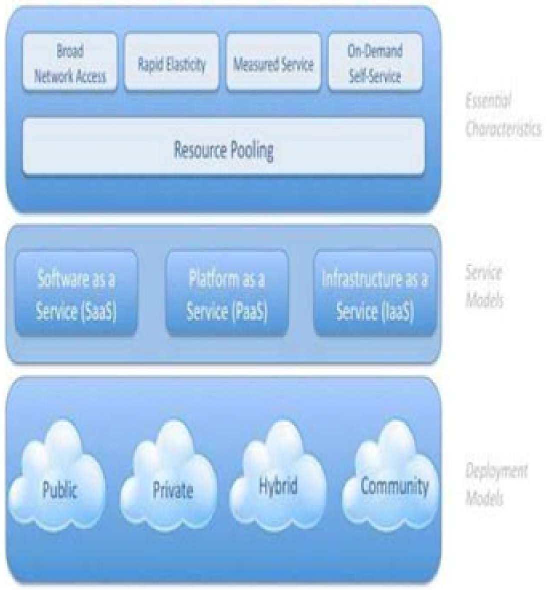
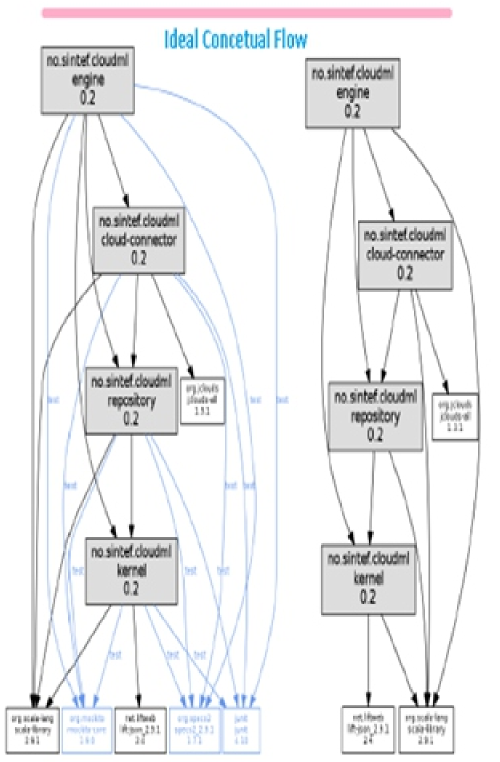


Fig.1.1 Showing the Summary view CloudEnvironment

2. Related work:

There are many cloud providers on the global market today. These providers support many layers of cloud, such as PaaS and IaaS. This vast amount of providers and new technologies and services can be overwhelming for many companies and small and medium businesses. There are no practical introductions to possibilities and limitations to cloud computing, or the differences between different providers and services. Each provider has some kind of management console, usually in form of a web interface and API. But model driven approaches are inadequate in many of these environments. UML diagrams such as deployment diagram and component diagram are used in legacy systems to describe system architectures, but this advantage has yet to hit the mainstream of cloud computing management.

It is also difficult to have co-operational interaction on a business level without using the advantage of graphical models. The knowledge needed to handle one provider might differ to another, so a multi-cloud approach might be very resource-heavy on competence in companies. The types of deployment resources are different between the providers; even how to gain access to and handle running instances might be very different.Although in reality end users are looking for something more than just creatinginstances in the cloud, they eventually want to move their application to the cloudenvironment. And to handle this they want a library which does as much of thisas possible, from pressing a button to having an application up and running on thecloud.


Ideal Concetual Flow
no.sintef.cloudml engine 0.2
no.sintef.cloudml cloud-connector 0.2
no.sintef.cloudml repository 0.2
no.sintef.cloudml kernel 0.2

(a) Dependency-graph with test-scope. (b) Dependency-graph without test-scope.

Fig.2.1 Dependency Graph based Analogy to put forward the Best Approach

In the above fig. 2.1 the benefit of a topology where the application is distributed over several nodes is the scalability and modularity, which were lacking in the single-node topology. For instance, if the user demand should rapidly increase, we need part ion based approach.

3. Methodology:

In the Public cloud environment, we need to consider the aspect of technical consideration where we need to take the technology based security mechanism along with performance where we can provide the best of technology. For scalability and modularity the single-node approach is restraining, i.e., it does not scale very well, and does not benefit from cloud advantages. If the application consumes too much CPU power, this slows the application totality down and decreases usability. There is no strong link between Cloud and the application, but to maintain scalability some measures must be manually developed into high level solution. So the initial application code includes support for work load distribution through application design and deployment considerations. In these measures consists of manually setting physical database address before deploying the application.

Multi-cloud:

Once able to provision the correct amount of nodes with desired properties on the first provider it became clear that mirroring the setup to the other provider is not as convenient as anticipated. There are certain aspects of vendor lock-in, so each script is hand-crafted for specific providers. The most noticeable differences would be (i) different ways of defining instance sizes, (ii) different versions, distributions or types of operating systems (images), (iii) different way of connection to provisioned instances. The lock-in situations can in many cases have financial implications where for example a finished application is locked to one provider and this provider increases tenant costs. Or availability decreases and results in decrease of service uptime damaging revenue.

Reproducibility:

The scripts provisioned nodes based on command-line arguments and did not persistthe designed topology in any way. This made topologies cumbersome to reproduce. If the topology could be persisted in any way, for example serialized files, it would be possible to reuse these files at a later time. The persisted topologies could also be reused on other clouds making a similar setup at another cloud provider, or even distribute the setup between providers.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Shareable:

Since the scripts did not remember a given setup it is impossible to share topologies “as is” between co-workers. It is important that topologies can be shared because direct input from individuals with different areas of competence can increase quality. If the topology could be serialized into files these files could also be interpreted and loaded into different tools to help visualizing and editing.

Robustness:

There are several ways the scripts could fail and most errors are ignored. They are made to search for specific lines in strings returned by the APIs, if these strings are non-existent the scripts would just continue regardless of complete dependency to information within the strings. A preferable solution to this could be transactional behavior with rollback functionality in case an error.In the view point of the load balance factor , we put search based top down approach having the mapping based on the key –value pair where hash based node approach in the big data analytics has been used to provide the best solution where we have to look forward for the best approach in the sense of high portability solution.

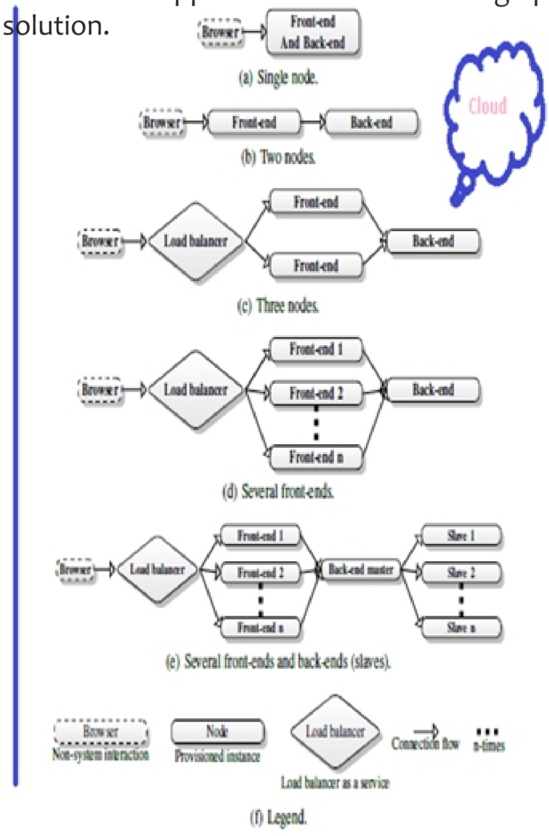


Fig.3.1 Architectural Illustration of Load Balance in the Connection of Cloud

Load balancer.
The feature of load balancer is implemented in cloud-engine, but at writing moment, is not supported by all clouds.

```
{
“name”: “test”,
“loadBalancer”: {
“name”: “test”,
“protocol”: “http”,
“loadBalancerPort”: 80,
“instancePort”: 80
},
“nodes”: []
}
```

After provisioning, or even deployment, is complete the users are left for themselves to manage any instances related to a provisioning. Such management could be to (i) terminate nodes, (ii) stop nodes, (iii) restart nodes, (iv) apply vertical scaling or (v) assign load balancer. The web-based consoles offered by providers are of a high standard. Because of this quality in consoles, there is not a greater problem for users to do management themselves. Although such functionality could be built into Cloud, not just as an alternative, but as an improvement for users already relying on Cloud. The implementation, cloud-engine, possesses the ability to terminate nodes through a method call, provided a nodes ID. This functionality has been mainly used for testing purposes, but is fully operational and available. Although just providing the means of functionalities needed to fulfill management is not sufficient for Cloud, as a model-driven approach there must be a refined link between templates,

4. Conclusion & Future Work :

Technology has its own limitation and in atoner side we can call it as the spectrum of its next level of Innovation. Considering all the aspect which we put forwarding the paper where load balancing is the major factor give n in the architectural diagram where we put forward for the topology based or in the other technical terminology we can tell it as the connection based on the number of hits and pool used for the same where essential to most businesses and private citizens that the major governments adopt the cloud and create a cloud strategy that models the best possible hybrid mix with the highest standards for security and service. It is probably that any provider that does not comply with these standards will go out of business, and companies that do not adopt the cloud will not be competitive. In the Aspect of future implementation we look forward for frame based or can tell as tool based approach for load balancing which IT industry call as product.

Reference:

[1] P. Hunter, "Gathering clouds," Engineering & Technology, vol. 8, no. 2, pp. 78–81, Mar. 2013.

[2] B. Zwattendorfer, K. Stranacher, A. Tauber, and P. Reichstädter, "Cloud Computing in EGovernment across Europe," in Technology-Enabled Innovation for Democracy, Government and Governance, vol. 8061, A. Kő, C. Leitner, H. Leitold, and A. Prosser, Eds. Springer Berlin Heidelberg, 2013, pp. 181–195.

[3]P. Mell and T. Grance, The NIST definition of cloud computing, http://csrc.nist.gov/ publications/ nistpubs/800-145/SP800-145.pdf, 2012.

[4]MicrosoftAcademicResearch,Cloud computing, http://libra.msra.cn/Keyword/6051/cloud-computing?query=cloud%20computing, 2012.

[5]GoogleTrends,Cloudcomputing,http://www.google.com/trends/explore#q=cloud%20computing, 2012.

[6] N. G. Shivaratri, P. Krueger, and M. Singhal, Loaddistributing for locally distributed systems, Computer,vol. 25, no. 12, pp. 33-44, Dec. 1992.

[7] B. Adler, Load balancing in the cloud: Tools, tips and techniques, http://www.rightscale. com/info center/ whitepapers/Load-Balancing-in-the-Cloud.pdf, 2012

.

[8] Z. Chaczko, V. Mahadevan, S. Aslanzadeh, and C. Mcdermid, Availability and load balancing in cloud computing, presented at the 2011 International Conference on Computer and Software Modeling, Singapore, 2011

[9] K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, N. Nitin, and R. Rastogi, Load balancing of nodes in cloud using ant colony optimization, in Proc. 14th International Conference on Computer Modelling and Simulation (UKSim), Cambridgeshire, United Kingdom, Mar. 2012, pp. 28-30.

[10] M. Randles, D. Lamb, and A. Taleb-Bendiab, A comparative study into distributed load balancing algorithms for cloud computing, in Proc. IEEE 24th International Conference on Advanced Information Networking and Applications, Perth, Australia, 2010, pp. 551-556.

[11] A. Rouse, Public cloud, http://searchcloudcomputing. techtarget.com/definition/public-cloud, 2012.

[12] D. MacVittie, Intro to load balancing for developers —The algorithms, https://devcentral.f5.com/blogs/us/introto-load-balancing-for-developers-ndash-the-algorithms, 2012.

[13] S. Penmatsa and A. T. Chronopoulos, Game-theoretic static load balancing for distributed systems, Journal of Parallel and Distributed Computing, vol. 71, no. 4, pp. 537-555, Apr. 2011.

[14] D. Grosu, A. T. Chronopoulos, and M. Y. Leung, Load balancing in distributed systems: An approach using cooperative games, in Proc. 16th IEEE Intl. Parallel and Distributed Processing Symp., Florida, USA, Apr. 2002, pp. 52-61.