ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

Load Balancing in Large Scale Cloud system Using Multiple Load Balancer Technique in the Cloud

S.Ilayanila, Prof.V.Ponkodi

M.Phil Research Scholar, Thanthi hans roever college, Perambalur, Tamil nadu, India

Assistant Professor in Computer Application, Thanthi hans roever College, Perambalur, Tamil nadu, India

ABSTRACT : Cloud computing is emerging as a new paradigm of large scale distributed computing. Load balancing is one of the main challenges in cloud computing which required to distributed the dynamic workload across multiple nodes to ensure that no single node is over whelmed. it helps in optimal utilization of resource and hence in enhancing the performance of the system . In complex and large systems, there is a tremendous need for load balancing. For simplifying load balancing globally (e.g. in a cloud), one thing which can be done is, employing techniques would act at the components of the clouds in such a way that the load of the whole cloud is balanced. For this purpose, we are discussing three types of solutions which can be applied to a distributed system: honeybee foraging algorithm, a biased random sampling on a random walk procedure and Active Clustering. the goal of load balancing is to minimize the resource consumption using Multiple load balancer in the Large scale cloud system. to achieves global load balancing through local serve action ,achieves load balancing across all system node using random sampling of the system domain then optimizes the job assignment by connecting similar services by local re-wiring.to perform well as system diversity increase and does not increase throughput as system size increases.

KEYWORDS : Cloud computing ,Load balancing, Large scale cloud system.

I. INTRODUCTION

The real strength of the cloud lies in the ability to distribute your workload (and your code) over multiple nodes employing multiple technologies to create a high performance, highly scalable, highly available platform at a highly affordable price. Cloud Load Balancers makes all of this possible by programmatically routing traffic to and between your Cloud Servers and other infrastructural resources, allowing cloud architects to employ tremendous amounts of scale and redundancy that would be difficult and costly to achieve within a dedicated environment. In addition to being the traffic cop of the open cloud, we'll discuss several other features of Cloud Load Balancers that can increase the performance, stability and security in website.

II. RELATED WORK

2.1 Honeybee Foraging Algorithm

This algorithm is derived from the behavior of honey bees for find ing and reaping food. There is a class of bees called the forager bees which forage for food sources, upon finding one, they come back to the beehive to advertise this using a dance called waggle dance. The display of this dance, gives the idea of the quality or quantity of food and also its distance from the beehive. Scout bees then follow the foragers to the location of food and then began to reap it. They then return to the beehive and do a waggle dance, which gives an idea of how much food is left and hence results in more exploitation or abandonment of the food source. In case of load balancing, as the web servers demand increases or decreases, the services are assigned dynamically to regulate the changing demands of the user. The servers are grouped under virtual servers (VS), each VS having its own virtual service queues. Each server processing a request from its queue calculates a profit or reward, which is analogous to the quality that the bees show in their waggle dance. One measure of this reward can be the amount of time that the CPU spends on the processing of a request. The dance

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

floor in case of honey bees is analogous to an advert board here. This board is also used to advertise the profit of the entire colony. Each of the servers takes the role of either a forager or a scout. The server after processing a request can post their profit on the advert boards with a probability of pr. A server can choose a queue of a VS by a probability of px showing forage/explore behavior, or it can check for advertisements (see dance) and serve it, thus showing scout behavior. A server serving a request, calculates its profit and compare it with the colony profit and then sets its px. If this profit was high, then the server stays at the current virtual server; posting an advertisement for it by probability pr. If it was low, then the server returns to the forage or scout behavior.

2.2 Biased Random Sampling

Here a virtual graph is constructed, with the connectivity of each node (a server is treated as a node) representing the load on the server. Each server is symbolized as a node in the graph, with each in degree directed to the free resources of the server. Regarding job execution and completion, Whenever a node does or executes a job, it deletes an incoming edge, which indicates reduction in the availability of free resource. After completion of a job, the node creates an incoming edge, which indicates an increase in the availability of free resource. The addition and deletion of processes is done by the process of random sampling. The walk starts at any one node and at every step a neighbor is chosen randomly. The last node is selected for allocation for load. Alternatively, another method can be used for selection of a node for load allocation, that being selecting a node based on certain criteria like computing efficiency, etc. Yet another method can be selecting that node for load allocation which is under loaded i.e. having highest in degree. If b is the walk length, then, as b increases, the efficiency of load allocation increases. We define a threshold value of b, which is generally equal to log n experimentally. A node upon receiving a job, will execute it only if its current walk length is equal to or greater than the threshold value. Else, the walk length of the job under consideration is incremented and another neighbor node is selected randomly. When, a job is executed by a node then in the graph, an incoming edge of that node is deleted. After completion of the job, an edge is created from the node initiating the load allocation process to the node which was executing the job. Finally what we get is a directed graph. The load balancing scheme used here is fully decentralized, thus making it apt for large network systems like that in a cloud.

2.3 Active Clustering

Active Clustering works on the principle of grouping similar nodes together and working on these groups. The process involved is: A node initiates the process and selects another node called the matchmaker node from its neighbors satisfying the criteria that it should be of a different type than the former one. The so called matchmaker node then forms a connection between neighbors of it which is of the same type as the initial node. The matchmaker node then detaches the connection between itself and the initial node.

III. EXISTING SYSTEM

Load balancing is the technique of distributing the load between various resources in any system. Thus load require to be distributed over the resources in cloud-based architecture, so that each resources does almost the equal amount of work at any point of time. Basic requirement is to provide some techniques to balance requests to provide the solution of fast response for request. Cloud Load Balancers manage online traffic by distributing workloads between multiple servers and resources automatically. They maximize throughput, minimize response time, and avoid overload. every load balancing algorithm must be such as to instate the required target. Some algorithms target at achieve higher throughput, some target at achieve minimum response time, some other target to achieve maximum resource utilization while some target at achieve a trade-off between all these metrics.

IV. PROPOSED SYSTEM

Load balancing in the cloud differs from classical thinking on load-balancing architecture and implementation by using commodity servers to perform the load balancing. From the Right scale website Patrick McClory Solutions Architect at RightScale he discussed in his article DNS Load Balancing and Using Multiple Load Balancers in the Cloud Load balancing in general is a complicated process, but there's some secret sauce in managing DNS along with multiple load balancers in the cloud. It requires that you draw from a few different sets of networking and “cloudy” concepts. I'll explain how to set up load balancers to build a fault-tolerant, highly available web application in the cloud. Here's what

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

you'll need: Multiple A records for a host name in the DNS service of your choice .Multiple load balancers to protect against failure Before I explain how the two work together, let's check out how each of them works individually.

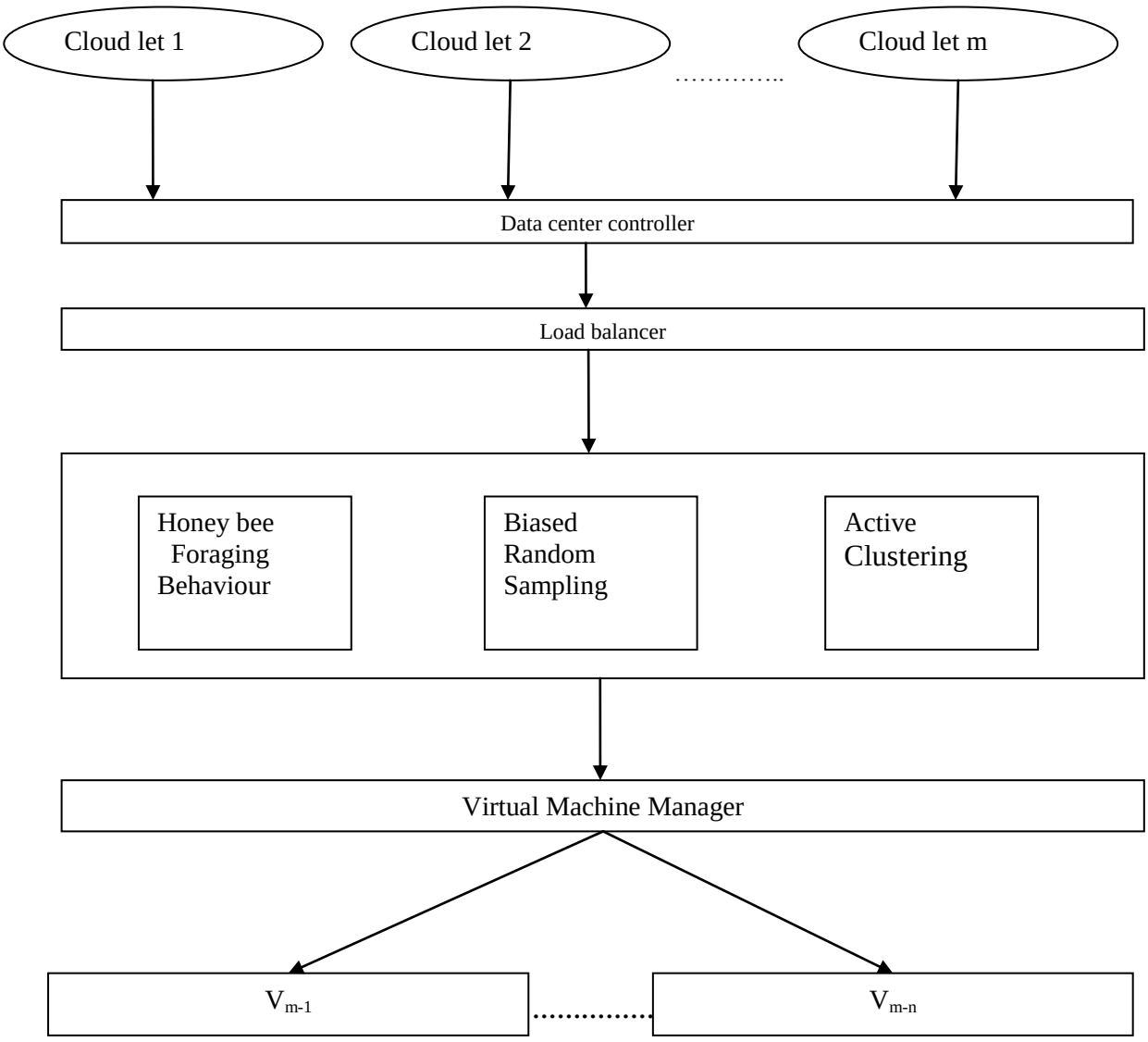

Figure 1 : Represent a framework underneath which various load balancing algorithms work in a Large scale environment

4.1 Multiple A Records for a Host Name

A records translate friendly DNS names to an IP address. For example, when you type rightscale.com in your browser, behind the scenes your computer is asking a DNS server to translate the name to an address. I'm working from a Mac and the process is a little different for Windows-based machines, so check out more on ns lookup for *nix and Windows. I'm also using one of Google's public DNS servers to perform my lookups. Check out the request below and note that when I query DNS, I'm getting a single address back for my test domain of dnsdemo.cloudlord.com:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

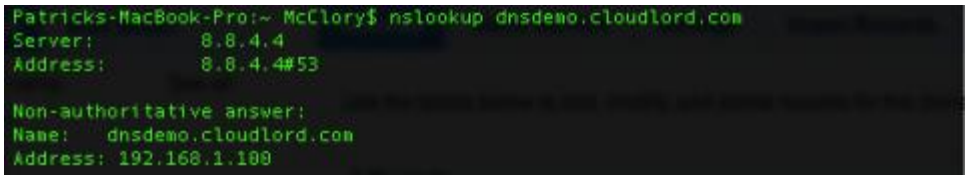

Figure 2 - DNS record with a single A record

My test domain has one A record associated and it resolves to the IP noted. Let's check out a more complicated example — Google.com:

Figure 3 - DNS lookup of google.com showing six A records

Note that as shown in Figure 4 below, Google returned six addresses (at the time I queried it) because Google has six A Records registered to serve its main domain. When I ran the same nslookup query again, the IP addresses were returned in a different order. This is commonly referred to as "DNS load balancing."

Figure 4 - DNS lookup of google.com showing A records in a different order than Figure 3

Figure 5 below shows DNS load balancing in action using dnstest.cloudlord.com and a test file that indicates which server is being served up. For this example, I set up my dnstest.cloudlord.com domain with two A records. Note that this first request has one attempt and the content reads that "this is server 1."

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

```
Patricks-MacBook-Pro:~ McClory$ curl -v http://dnstest.cloudlord.com/index.html
* About to connect() to dnstest.cloudlord.com port 80 (#0)
*   Trying 23.23.164.125...
* connected
* Connected to dnstest.cloudlord.com (23.23.164.125) port 80 (#0)
> GET /index.html HTTP/1.1
> User-Agent: curl/7.24.0 (x86_64-apple-darwin12.0) libcurl/7.24.0 OpenSSL/0.9.8
r zlib/1.2.5
> Host: dnstest.cloudlord.com
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/html
< Last-Modified: Fri, 19 Oct 2012 02:36:25 GMT
< Accept-Ranges: bytes
< ETag: "7fe18d88a2adcd1:0"
< Server: Microsoft-IIS/7.5
< Date: Fri, 19 Oct 2012 02:36:59 GMT
< Content-Length: 16
<
* Connection #0 to host dnstest.cloudlord.com left intact
this is server 1* Closing connection #0
```

Figure 5- Curl request to test domain with full availability of all servers

Next, I terminated the first server on the next request (to force a failed state), and the results are shown in Figure 5 below. Note that there's a timeout on the first IP, and then the second request goes through without any issue. You'll also notice that it's returning a response of "this is server 2."

```
Patricks-MacBook-Pro:~ McClory$ curl -v http://dnstest.cloudlord.com/index.html
* About to connect() to dnstest.cloudlord.com port 80 (#0)
*   Trying 23.23.164.125...
* Operation timed out
*   Trying 23.23.168.195...
* connected
* Connected to dnstest.cloudlord.com (23.23.168.195) port 80 (#0)
> GET /index.html HTTP/1.1
> User-Agent: curl/7.24.0 (x86_64-apple-darwin12.0) libcurl/7.24.0 OpenSSL/0.9.8
r zlib/1.2.5
> Host: dnstest.cloudlord.com
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/html
< Last-Modified: Fri, 19 Oct 2012 02:35:33 GMT
< Accept-Ranges: bytes
< ETag: "b65e369a2adcd1:0"
< Server: Microsoft-IIS/7.5
< Date: Fri, 19 Oct 2012 02:39:30 GMT
< Content-Length: 16
<
* Connection #0 to host dnstest.cloudlord.com left intact
this is server 2* Closing connection #0
```

Figure 6 - Curl request to test domain with primary A record in failed state (note timeout and new IP)

The order in which IP addresses are returned varies by the DNS server and provider used but often follows a round-robin or geographically specific algorithm. The idea here is that different clients will get different ordered lists of IP addresses corresponding to your domain name. This has the effect of distributing requests across the group of IPs in a specific manner. If an IP address is does not respond in an appropriate amount of time, the client will time out on that request and move on to the next IP address until the list is exhausted or it finds a connection that's valid. Although it's not an exhaustive list, most modern browsers, along with curl as shown above in Figure X, follow this retry process. There are a few things to remember though: DNS failover doesn't provide any additional features such as “sticky sessions” for your application. Upstream DNS caching is unpredictable -client DNS providers may or may not respect your TTL settings. This isn't a replacement for TCP load balancing because it's not terribly precise based on the upstream DNS caching process noted above.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

4.2 Multiple Load Balancers for Redundancy and Scalability

With multiple IP addresses routing to your deployment, each of these addresses can terminate at a load balancer that serves your back-end application (see Figure 7 below). Doing this, you'll be able to present multiple endpoints to the public to serve your application (I'll get back to why this is important in a minute).

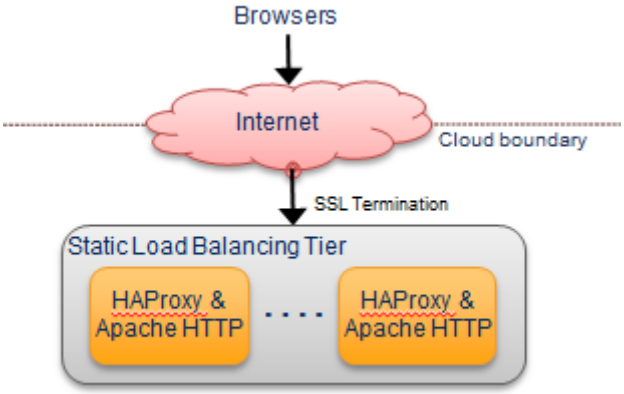

Figure 7- Overview of incoming connections from public Internet to TCP load balancers

In Figure 8 below, I go a step further and illustrate how connectivity to the application layer can be set up from multiple TCP load balancers. This allows you to have multiple incoming connections each serving up the same content, providing a redundant load balancing layer as well as a redundant application layer.

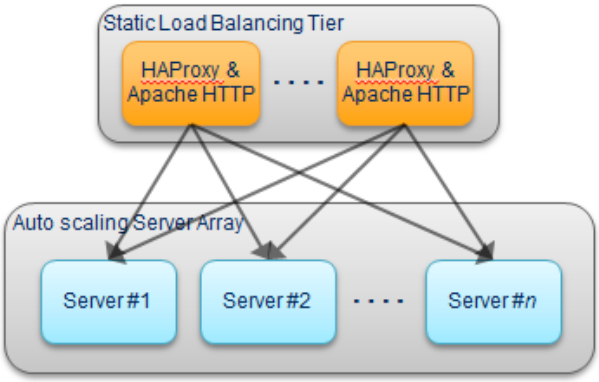

Figure 8- Connection diagram for multiple load balancers connecting redundantly to the same application server tier

DNS Load Balancing: Bringing It All Together The end result is that by using DNS load balancing, you can achieve a fairly rough balance of traffic between multiple TCP load balancers, which can manage applying load to your application servers at a more granular level:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

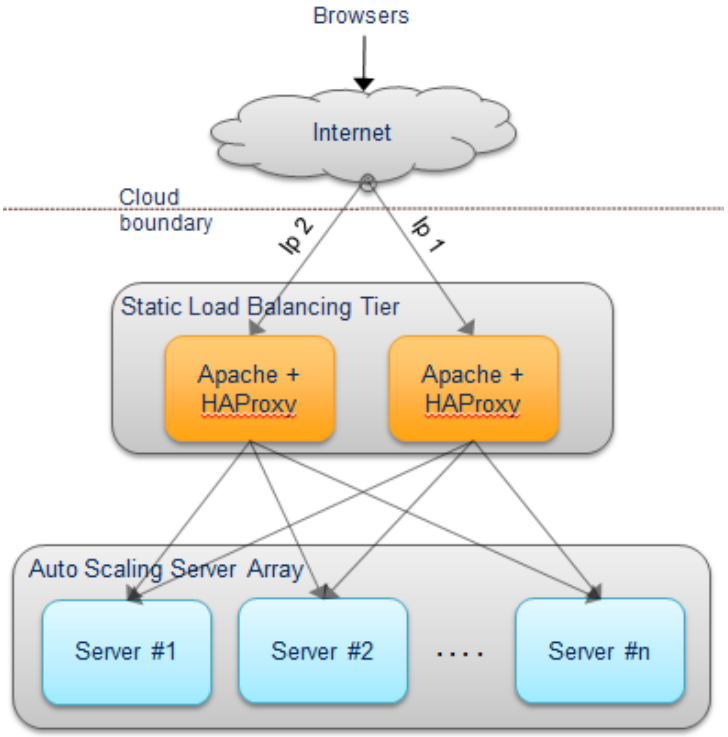

Figure 9 - Full incoming connection diagram showing multiple load balancers with their own IP address

This is a great way to protect against failure and increase overall throughput, giving you the ability to scale for high availability and high performance. For more information on metrics related to configuration and throughput on HAProxy in the cloud, check out this white paper, Load Balancing in the Cloud: Tools, Tips and Techniques. Setting up DNS load balancing can be a bit of a hassle, but the Load Balancer with HAProxy ServerTemplate, along with scripts for application servers to attach to load balancers, simplifies the process. The RightScale Server Template™ and scripts use a tag-based, managed solution that will keep your HAProxy config files synchronized and that will automate the deployment, registration, and detach process for all servers involved.

V. CONCLUSION AND FUTURE WORK

Our objective for this paper is to develop an effective load balancing algorithm using Multiple load balancing technique to maximize or minimize different performance parameters like CPU load, Memory capacity, Delay or network load for the clouds of different sizes. In this paper, Large scale cloud system based on Honeybee foraging behavior ,biased random sampling and active clustering algorithm are used in cloud computing architecture. This modification supports to minimize the make span of the cloud computing based services and portability of servicing the request also has been converged using the combination of technique. This technique does not consider the various environment issues. Researchers can proceed to include the cloud storage, Three level cloud computing network issues in their future researches.

REFERENCES

1. A. Khiyaita, M. Zbakh, H. El Bakkali, and D. El Kettani, "Load balancing cloud computing: state of art," in Network Security and Systems (JNS2), 2012 National Days of, pp. 106–109, IEEE, 2012.
2. N. J. Kansal and I. Chana, "Existing load balancing techniques in cloud computing: A systematic review.," Journal of Information Systems & Communication, vol. 3, no. 1, 2012.
3. D. A. Menasc´e and P. Ngo, "Understanding cloud computing: Experimentation and capacity planning," in Computer Measurement Group Conference, 2009.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 2, February 2015

4. S.-C. Wang, K.-Q. Yan, W.-P. Liao, and S.-S. Wang, "Towards a load balancing in a three-level cloud computing network," in Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, vol. 1, pp. 108–113, IEEE, 2010.
5. T. D. Braun, H. J. Siegel, N. Beck, L. L. B¨ol¨oni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, D. Hensgen, et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," Journal of Parallel and Distributed computing, vol. 61, no. 6, pp. 810–837, 2001.
6. T. Kokilavani and D. Amalarethinam, "Load balanced min-min algorithm for static meta-task scheduling in grid computing.," International Journal of Computer Applications, vol. 20, no. 2, 2011.
7. T. Anandharajan and M. Bhagyaveni, "Co-operative scheduled energy aware load-balancing technique for an efficient computational cloud.," International Journal of Computer Science Issues (IJCSI), vol. 8, no. 2, 2011
8.J. M. Galloway, K. L. Smith, and S. S. Vrbsky, "Power aware load balancing for cloud computing," in Proceedings of the World Congress on Engineering and Computer Science, vol. 1, pp. 19–21, 2011.
9. R. Alonso-Calvo, J. Crespo, M. Garcia-Remesal, A. Anguita, and V. Maojo, "On distributing load in cloud computing: A real application for very-large image datasets," Procedia Computer Science,vol. 1, no. 1, pp. 2669–2677, 2010.

BIOGRAPHY

S.ILAYANILA ,M.Phil Research Scholar, Thanthi hans roever college,Perambalur. Her area of interest is in cloud computing.

Prof.V.Ponkodi,MCA.,M.Phil.,(Ph.D).,Assistant Professor in Computer Application, Thanthai Hans Roever college,Perambalur-621212.