Round Robin Approach for Load Balancing in Cloud Computing

Akhil Karrothu (ak8367), Sri Rachana Achyuthuni (sa9547)

1 Background

When hosting any application, each web application might receive several hundreds of requests. Whenever this happens, it is important that every request is responded to, in a timely manner. If all requests are sent to a single server or a couple of servers, there is a high chance that they will not respond. This is because each server will have certain thresholds in terms of CPU Utilization, Memory, etc. Systems might crash or become unresponsive when these thresholds are met. Thus, it is important to make sure that each system is not over worked. In order to maintain equal load amongst the cluster of systems hosting a web application, a Load Balancer is often deployed as the door to this cluster. A Load Balancer is nothing but a system with an algorithm to determine the system to which requests need to be redirected and some proxy application implemented to redirect the requests.

A Load Balancer is often used hand in hand with an AutoScaling system which scales the systems in the clusters based on the existing load, the status of the systems within the cluster or based on any other fixed parameters such as the time of the day or the day of the week.

2 Approach

Round Robin Algorithm is a widely used algorithm for scheduling or for distributing resources.[6] In our project, we will be implementing Round Robin Algorithm to distribute requests to each server for Load Balancing in Cloud Computing. In Round Robin model, the requests should be sent to each server in circular order, in an equal manner. In this approach, we would be using an EC2 server[2] as a Load Balancer along with an Amazon AutoScaling[1] Group having a cluster of EC2 servers. We will host a small website for our testing purposes using NodeJs. We will also be using a proxy server, Nginx [5], for redirection of requests.

After implementing this algorithm, we will be examining it closely for any drawbacks and limitations. We will then report these limitations and also, propose solutions to fix these limitations.

3 Expected Results

We should be able to view the load balanced application and the requests being sent to servers in Round Robin fashion, successfully. We should also be able to see any shortcomings in this redirection process.

4 Comparative Analysis

We will be comparing the approach that we proposed and implemented with some of the existing approaches for Load Balancing such as AWS Classic Load Balancer [3] and HAProxy [4].

5 Time Line

- 1stSeptember - 15thOctober: Build the algorithm.
- 15thOctober - 23rdOctober: Host an application, implement the algorithm and obtain the results.
- 23rdOctober - 7thNovember: Examine and obtain and limitations and also, come up with possible solutions for fixing them.
- 7thNovember - end of semester: Implement other established Load Balancing techniques and present a comparative analysis.

References

[1] *Amazon AutoScaling*. https://aws.amazon.com/autoscaling/. Accessed: 2020-09-04.

[2] *Amazon EC2*. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html. Accessed: 2020-09-04.

[3] *AWS Classic Load Balancer*. https://aws.amazon.com/elasticloadbalancing/. Accessed: 2020-09-04.

[4] *HAProxy*. https://aws.amazon.com/elasticloadbalancing/. Accessed: 2020-09-04.

[5] *Nginx*. https://www.nginx.com. Accessed: 2020-09-04.

[6] *Round Robin Algorithm*. https://en.wikipedia.org/wiki/Round-robin_scheduling. Accessed: 2020-09-04.