Report for the OpenStack Cloud Service Deployment and Operation project

Hailay Gebreslasie Gebremeskel
Department of Computer Science
Blekinge Institute of Technologyu
hage21@student.bth.se

I. Design

In this project, I have designed a solution to deploy and operate a service within an OpenStack Cloud. The design consists of five nodes: one proxy server, one bastion server, and three worker nodes.

The proxy server is responsible for load balancing traffic entering the network. I have chosen to use NGinx as the load balancer because it can handle both TCP and UDP-based traffic. This ensures efficient distribution of requests to the worker nodes. To avoid a single point of failure and improve the overall availability of the service, two proxy servers are deployed. The proxy servers are grouped in to a single virtual server sung keepalived software. The two servers are configured in active-standby mode.

The bastion host acts as an entry point to the internal network. It provides secure access for authorized users to connect to the internal resources. This enhances the security of the solution by limiting direct access to the internal network and allowing only authorized SSH connections through the bastion host

The worker nodes are responsible for providing the service. These nodes can dynamically scale based on demand. By adding or removing worker nodes, we can adjust the capacity of the service to handle varying loads. This scalability ensures that the solution can accommodate increasing traffic and effectively utilize resources.

To connect the private network to the external network, a router is included in the design. The router enables communication between the internal network and the external network, allowing access to the service from the outside.

Overall, this design provides scalable solution for deploying and operating a service within an OpenStack Cloud by considering network range 10.0.1.1/24

II. Performance Evaluation

To evaluate the performance impact of varying the number of nodes, an Apache Benchmark (ab) test was conducted. The ab requests were sent to the public IP of the proxy server. The test configuration consisted of 10,000 requests with 10 concurrent connections for each number of nodes (1, 2, 3, 4, and 5).

For each number of nodes in the system, a request is sent 5 times, and then the average mean and standard deviation is calculated. This is done for more accurate measurement of the system's performance. By repeating the request multiple times, any variations or outliers in response times can be minimized, and a more representative average value can be obtained.

A. Performance Evaluation for one node

Sending the ab test to the public IP of the proxy server for one node displays a result as in $table_1$ below.

TABLE I
Performance for one node

Metric	Mean	+/-sd
Connect	46.4	10.88
Processing	43.8	8.3
Waiting	43.4	7.358
Total	90.4	14.16

B. Performance Evaluation for two nodes

Sending the ab test to the public IP of the proxy server for two nodes display a result as in $table_2$ below.

TABLE II
Performance for two nodes

Metric	Mean	+/-sd
Connect	45	6.38
Processing	43.2	8.3
Waiting	42.4	7.33
Total	88.2	10.82

C. Performance Evaluation for three nodes

Sending the ab test to the public IP of the proxy server for three nodes display a result as in $table_3$ below.

D. Performance Evaluation for four nodes

Sending the ab test to the public IP of the proxy server for four nodes display a result as in $table_4$ below.

TABLE III
PERFORMANCE FOR THREE NODES

Metric	Mean	+/-sd
Connect	44.4	12.54
Processing	42.6	6.62
Waiting	42	5.94
Total	87.2	14.38

TABLE IV
PERFORMANCE FOR FOUR NODES

Metric	Mean	+/-sd
Connect	46.4	10.88
Processing	43.8	8.1
Waiting	43.2	7.58
Total	90.2	14.16

E. Performance Evaluation for five nodes

Sending the ab test to the public IP of the proxy server for five nodes display a result as in $table_5$ below.

The collected results were analyzed, focused on mean and standard deviation (sd) values. These metrics provide insights into the response time and consistency of the system under different node configurations. To ensure statistical significance,

TABLE V
PERFORMANCE FOR FIVE NODES

Metric	Mean	+/-sd
Connect	46	15.86
Processing	44	14.62
Waiting	43	13.42
Total	90.6	22.18

multiple iterations of the test were performed, and the results were averaged.

III. LARGE-SCALE OPERATION

The solution presented is designed to efficiently handle a large number of users, scaling both in terms of network and service capacity. By implementing load balancing techniques, the solution can effectively distribute incoming traffic across all available worker nodes. Du to the limit in flavor, this architecture allows creating up to 7 worker nodes without any modification which could serve thousands of users without significant performance degradation.

However, in order to accommodate an even larger user base, modifications to the architecture are necessary. To handle a substantial increase in traffic, additional worker nodes with sufficient CPU and RAM resources would need to be provisioned. This would require expanding the network range to accommodate the increased number of nodes. It is important to note that such modifications would require a complete redesign of the solution from scratch, as changing the network infrastructure is a complex and involved process.

Considerations for scaling the solution to handle an extremely large number of users include:

Worker node provisioning, network expansion, redesign and reimplementation and resource management.

Potential problems that may arise when scaling the solution include:

- Increased Latency: As the number of users grows, the response time may increase
- Network Congestion: With a larger user base, network congestion may occur, impacting the overall network performance and causing delays in data transmission.
- Maintenance and Upkeep: Managing a large-scale infrastructure requires diligent maintenance and proactive monitoring to address potential issues and ensure smooth operation.

Operating the service from multiple locations may add additional challenges related to networking and service performance such as:
Data synchronization, latency and load balancing across multiple locations.

Finally, scaling the solution to handle a large number of users and operating from multiple locations requires careful planning, architectural modifications, and addressing potential challenges specific to network and service performance.