The article discusses the design and evaluation of a distributed system in multiple clouds using Docker Swarm cluster. Despite the popularity of multi cloud infrastructure, due to its implementation challenges it is cumbersome to design distributed systems in multiple clouds. Virtualization is the base to implement this system but because of its limitations Docker Swarm, a container-based clustering tool came into the picture which amalgamated several inbuilt attributes of the distributed system.

The paper sheds a light on the docker which is an open-source platform for the development of software's using containers and how it supports the design of multi cloud distributed systems. It presents a simulated development of the system in VirtualBox, highlighting attributes such as high availability, fault tolerance, scalability, load balancing, and maintainability. The evaluation demonstrates that Docker Swarm is easy to design and replicable in multiple cloud environments.

Furthermore, jargons related to docker like containerization, containers and finally Docker Swarm is discussed in Section II of the paper. Containerization is an OS level virtualization-based technique where we create an isolated environment like virtual machine but without hardware. Docker container is an instance of containerization. If a docker container is used on Linux then Linux OS acts as a default OS host, whereas in other OS's separate Docker host needs to be installed. Docker host is a lightweight VM compared to actual VM. Docker Swarm is a container orchestration tool that controls several docker containers or nodes to form a single virtual system. Because of its availability, reliability, scalability, fault tolerance and maintainability it is a great tool for the cluster management.

Section III clarify the design of a distributed system using Docker Swarm, which is supported by all five clouds. The design consist of three manager nodes and two worker nodes deployed on multiple clouds. The manager coordinates and ensure high availability by electing a leader if the current leader fails. Workers managed by managers perform the elementary operations of executing tasks and routing data for containers. Managers ensure regular health check of worker nodes, as a result this design is highly available and reliable distributed system.

Furthermore, in Section IV, the authors simulated the distributed system design illustrated in Section III inside the VirtualBox using Docker Swarm. The simulation consists of five docker machines, three swarm managers and two swarm workers having assigned private IP addresses and standard docker port. The process of creating the Docker Swarm-based distributed system is easy and can be replicated on multiple clouds.

Section V evaluates the simulated distributed system design based on high availability and fault tolerance, automatic scalability, load balancing and maintainability of services. This simulation demonstrated that the system could run on threshold of two manager nodes in cases of failure of any manager node. This proves that Docker-Swarm based distributed system can offer high availability and fault tolerance for one failure when it has three active managers. A Docker Swarm-based distributed system provides automatic scalability, load balancing, and maintainability of services. In the evaluation of these attributes, two services (nginx and Redis) are created on the Docker Swarm cluster and are scaled to five instances. Docker Swarm automatically balances the load across the cluster and assigns replicas to nodes with the best available resources. The maintainability of the system is tested by abruptly stopping two nodes, and it is observed that the instances of the services running on those nodes are automatically switched to other running nodes, ensuring uninterrupted service availability. Finally, the Docker Swarm has been compared with other orchestration tool like Kubernetes on the terms of scalability for the large cluster size. With the help of research, it was evident that the Docker Swarm was, on average, five times faster in terms of container Startup time and seven times faster in terms of system responsiveness compared to Kubernetes. However, it was concluded that both frameworks are different in attributes, and each will perform better under certain circumstances, Docker swarm for small scale workloads and Kubernetes for large scale workloads.

To conclude, paper evaluated a Docker Swarm-based distributed system, exemplified its ease of design & performance attributes, and proposed future evaluation on larger clusters & multiple cloud environments.

Critique

The evaluation of the distributed system appears to be limited to a simulated environment. While this provides some insights into the system's behavior, it would be valuable to conduct evaluations in real-world scenarios with larger clusters and multiple cloud environments to assess scalability, performance, and potential challenges.

Improvements

Real-world Use Cases: Including real-world use cases or scenarios where the proposed approach has been implemented would add practical relevance to the article. This could involve describing specific industries or applications where Docker Swarm-based distributed systems have been successfully deployed.

References

[1] N. Naik, "Performance Evaluation of Distributed Systems in Multiple Clouds using Docker Swarm," 2021 IEEE International Systems Conference (SysCon), Vancouver, BC, Canada, 2021, pp. 1-6, doi: 10.1109/SysCon48628.2021.9447123.