JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

DYNAMIC WEB SERVER BASED ON OPERATING SYSTEM-LEVEL VIRTUALIZATION USING DOCKER STACK

[1]Kalpana Ettikyala, [2]Dr. Y Vijayalata
[1]Assistant Professor, [2]Professor
[1]CSE Department
[1]CBIT, Gandipet, Hyderabad, India. Email: kalpana_e@cbit.ac.in

Abstract: The increasing wishes of the community for virtual records are very high. A dynamic internet server is wanted with the aid of a website to fulfill the unexpectedly growing wishes of consumers to get entry to. The incapability of the physical server to receive requests from customers in huge numbers makes the internet site hard to get admission to. Operating machine-degree virtualization is extensively used because it can overcome problems that arise on physical servers. This affects the extent of availability in coping with an excessive variety of requests. The weight balancing server mechanism is used to deal with the problem, one in every of them is through using the assistance of the Docker tools. For excessive availability, virtualization on the Docker helps the web server serve high requests with the aid of using a Docker swarm. Within the Docker swarm, there's a division of roles, particularly the roles of employees and bosses with different responsibilities. Proposed approach dynamic web server cluster using a Docker swarm can improve the excessive stage of availability of an internet server provider as the wide variety of packing containers that run on the swarm.

Index terms- **Docker, Web Server, Swarm Docker**

I. INTRODUCTION

A web server is a problem that identifies and provides comprehensive real-time resources via an Internet browser, with responses in the form of web pages and HTML files. The most prevalent source of Internet server downtime is the availability level. Call for impacts the full functionality of a web server that receives a lot of traffic, causing the web to turn upside down if the web server becomes too congested. The Docker Stack combines multiple services and allows them to run on a few connected machines. The answer is to use an active gadget-level visualization offered by a collecting system supplied by docker swarm, a dock stack used to distribute images with numerous containers and web server services to other digital servers.
The findings of this study show that the device-degree virtualization function facilitates failover and loading measurement methods within the docker swarm area, and the docker stack plays a role in distributing web server services from the node administrator to the node server as a backup web server service.

A) Problem Definition

Many websites were still up and running at the time, roaming the internet and being frequently used in product brochures and company biographies. However, many websites have begun to become dynamic, collaborative, and job-oriented in order to be employed in information systems, banking, and a variety of other applications. Changes in digital knowledge can be seen in the usage of many dynamic websites, as well as in the use of a strong web server.
The availability issue is the most serious issue that can arise on a web server. A flexible web can be utilized in this circumstance if the web server just delivers a few static web pages as simple information. The static web can be used, if necessary, to improve security systems or to connect with users in real time. There will be an increase in website access demands on the web server as the number of requests from information consumers increases at the same time. The application has an impact on the web server's performance since it may handle requests when the web server is overburdened. The website will crash and be unable to be shown if the web server is disrupted, leading other services to be unavailable. Incidents like these will harm web server service providers' availability.

B) Methodologies

Starting with the numerous uses of dynamic websites, it also has an impact on the use of the dynamic web server. In place of the cluster swarm, a load balancing system and a web server failover are required to support a web server that is efficient with real-time usage and high availability. One way that a web server's management can increase its performance is to use upload rating. To move load from one web server to another, you must first download it. To distribute the load among the web servers that is available. Also, make the website's availability status more realistic, and reduce the time it takes for the website to respond. Failover is a backup application in which a node's service, such as software or hardware, fails and is moved to another node in the existing system. This automated process will replace any services that are falling as before.
A swarm docker can be used to create failover deployments at the operating system level. Swarm docker is used to deploy flexible container-based web servers. The process of setting up a webserver is done separately for each host server in order to enable faster data transfers. With this technology, it is envisaged that the web server would operate more efficiently by using a swarm docker at the network virtualization level. With the support of a swarm manager, which is maintained by a swarm administrator, Docker Swarm can use load balancing to send web server resources to available hosts.

II. RELATED STUDY

With quick access to a user's digital information and a rise in website access requests, the web server will get more requests and will be unable to fulfill demands from users. Because the web server is overburdened with applications, it slows down. Access to the website is disrupted as a result of this problem. Those who use a web server service to host their site may suffer losses as a result of this. Because the website will be temporarily unavailable. Hardware upgrades on the server are undertaken to improve its performance in order to meet the full demand of users for web services. This technique, however, only addresses the issue of short-term availability[1]. One approach to use web server administration to increase performance is to load uploads. Load balancing is the process of moving traffic from one web server to another. To distribute the load among the web servers that are available[2]. Also, make the website's availability status more realistic, and reduce the time it takes for the website to respond. It is also vital to control the failover promptly when balancing the load. Failover is an automatic navigation system that can run a backup operation in the present system if one of the programs experiences a downtime[3].

A multi-level detection technique will be used to apply docker swarm to host collections. Docker swarm is a docker-provided container orchestration solution for distributing docker containers to the system collection. The webserver service in the cluster system will be dispersed based on system needs and will run in multiple portable containers built and installed in an existing swarm cluster. Under the control of a control docker called swarm manager[4], Docker swarm will also do load balancing to monitor server speed and stability. In this study, there is a detailed description of the implementation of the container-based web server using swarm docker. The webserver configuration process is done individually for each host server in order to enable dynamic data transfer. It is predicted that the web server would run more efficiently using swarm docker at the virtualization level network with the usage of this technology. With the help of a swarm administrator, Swarm can use load balancing to send web server resources to available hosts, which will be managed exclusively by the swarm administrator. Background research into the use of flexible web servers at the operating system level. A swarm docker is used for virtualization. The goal of this research was to address flaws in the deployment of load balancing on portable servers that were not meeting the availability factor in the case of an interruption[5]. Even the Strategy of Recognition-Availability may indicate higher service availability. Alternatively, by lowering the top to the first process, docker swarm can save nearly half of the final space[6]. Docker Swarm also includes a simple and efficient visual interface for managing, feeding, and distributing apps to Clusters. Containers may be employed in both private and public clouds[2], according to Docker swarm. In order to meet availability requirements, it may also modify the number of containers in a couple of seconds[7]. In a select circumstances, docker swarm can also lower the rate of delay and speed up service setup[8].

Virtualization based on the operating system refers to system performance aspects in which the kernel allows for the presence of several user environment conditions. Virtualization at the operating system level is a type of server virtualization in which the kernel runs the system, allowing numerous instances of a single user space, also known as software containers or containers. In virtual hosting environments, virtualization of the operating system level is commonly utilized to separate limited hardware resources from the most unreliable users. System administrators can use it again to integrate server hardware with delivery services for several host containers on a single server, but it will be slow. The operating system virtualization level implementation can migrate directly, but it can also balance the varying load of containers between nodes in the collection.

The use of a docker stack in a group can deploy and control powerful web server resources for optimal performance in installing web server services in each location and cluster system performance support. The availability or success of the system employing the web server service is improved by failures and detection mechanisms.

III. SYSTEM DESIGN AND IMPLEMENTATION

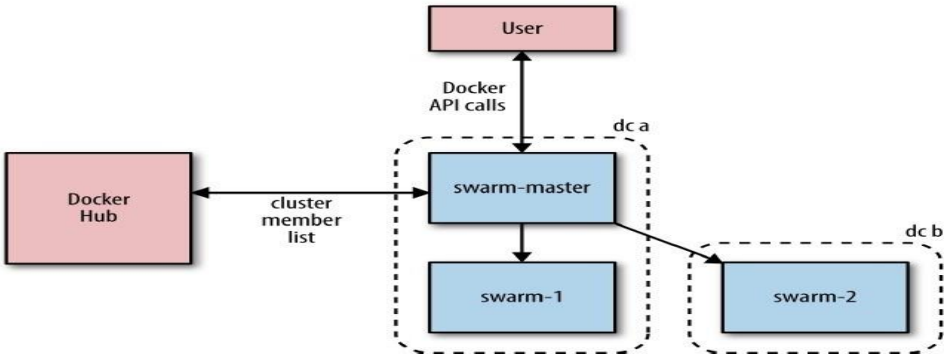


Figure-1: Block diagram of Docker Swarm

Docker Hub

A Swarm agent is installed on each host, and a Swarm manager is installed on one of the hosts (on small take a look at clusters this host may additionally run an agent). The manager is in charge of the packing container orchestration and scheduling at the hosts. Swarm can be configured to run in a high-availability mode, with etcd, Consul, or ZooKeeper handling failover to a backup manager.

In Swarm, there are a variety of ways for finding and delivering hosts to a cluster, which is referred to as discovery. By default, token-primarily based discovery is employed, in which the addresses of hosts are saved in a Docker Hub listing.

Discovery service

Within the back-ends of Docker Swarm, there are numerous Discovery services that are used to discover the nodes in the cluster utilizing a distributed key/value store, such as Consul, Etcd, or Zookeeper. A list of nodes or a static report. As a hosted discovery provider, Docker Hub. It also aids any modules that comply with the Discovery API interface.

Scheduler

During the development and execution of a container, the Docker Swarm scheduler decides which nodes to use. There are two steps to the scheduling process: To determine which nodes can be used, the scheduler first looks at the consumer's filters.

High availability of docker swarm manager

The Swarm Supervisor responds to the cluster and maintains the assets of several Docker Daemons that are served by Swarm Nodes. If the Swarm manager dies, a new one will be generated to deal with the provider's interruption. There are multiple Swarm managers in Docker Swarm's high availability feature. A machine supervisor has the ability to produce a number one manager as well as a few duplicate times. Requests can be robotically proxied to the number one manager if they are dispatched at different times. In addition, if the number one manager fails, the other duplicate times will result in the appointment of a new primary supervisor.

High availability of Docker Swarm containers

Docker Swarm has a reschedule policy that you can set at the start of a field. Whenever a swarm node fails, the swarm supervisor restarts all package containers in that swarm node for other active swarm nodes.

The system design defines the relationships and communication flows between the five components: the master node as an administrator, worker node 1, worker node 2, and the Docker volume as a data sharing point. The intermediate stream of components from the master node1 acting as an administrator uses the docker swarm init command to give tokens to workers, and then node1 and node2 acting as admins use the command to call docker swarm to join. node1 and node2 can participate in node1. This runs on several components: Node1, acting as an administrator, Node1, and Node2 acting as an administrator. The support feature of this program is the volume of the dock outside the area that acts as a file share or directory that shares a location within the desired dock container. This method is used for disk sharing by first inserting the disk into the Docker server. For each component in the system to function properly, a stable internet connection is required for each component to be fully connected. To be able to use the web server, you need to create an image created with a Dockerfile. The Dockerfile contains steps to install the tools needed to support the use of your web server.

Table-1: Node Manager IP Address

NAME	IP ADDRESS
Master Node	192.168.33.12
Worker Node 1	192.168.33.13
Worker Node 2	192.168.33.14

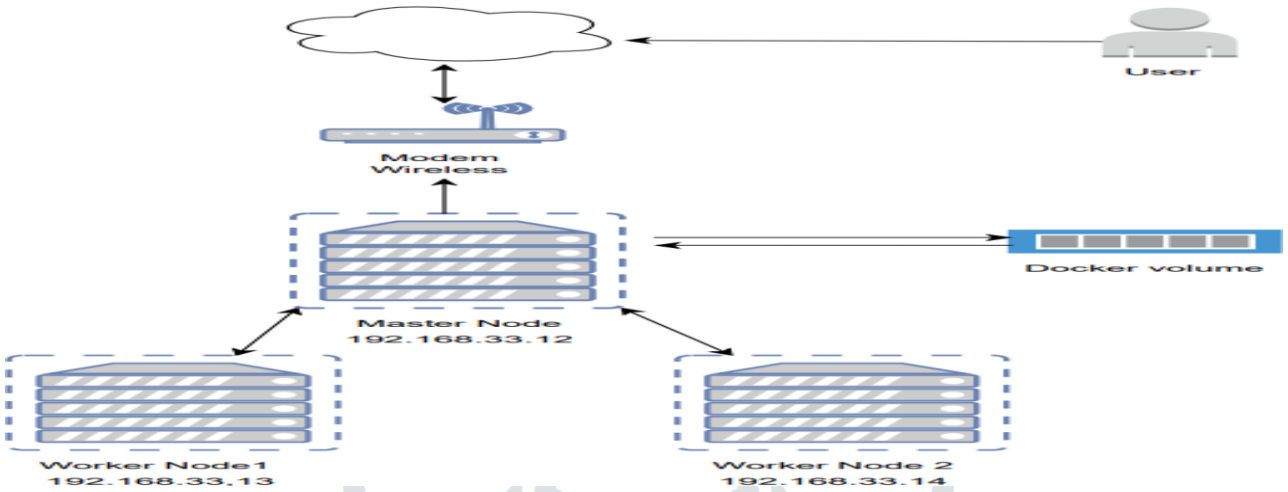
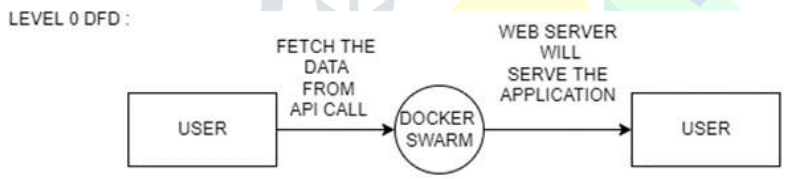
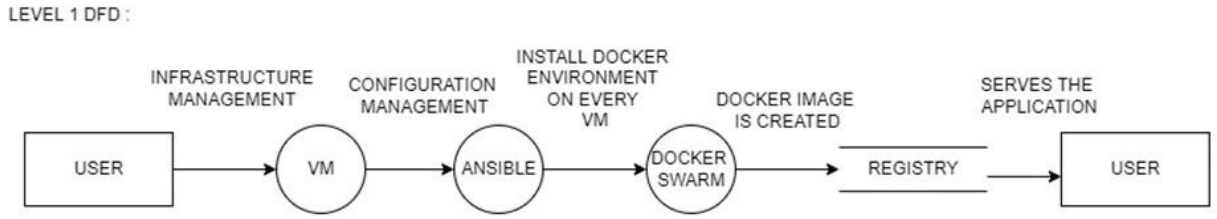
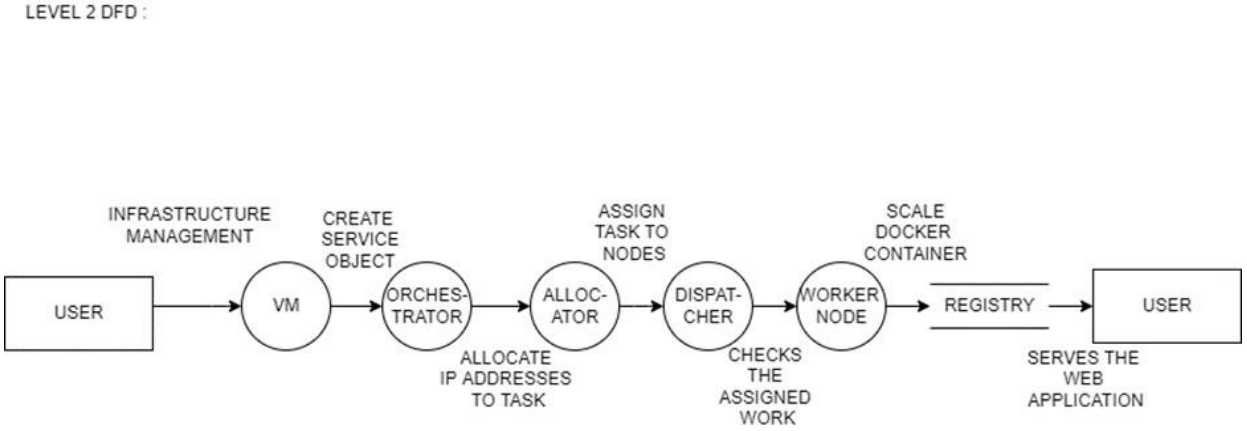


Figure-2: System design

The design of the system defines the flow of relationships once and for all network connectivity over the system as shown in Figure-2. This program needs 3 nodes / hosts that have their own requirements or roles. The node will be run in a virtual machine environment using the virtual box.







The dynamic web server launcher uses Docker Swarm. The implementation is based on what is described in the system design, according to the requirements needed to be able to solve the problems that exist in the physical web server. Optimize and manage powerful web server resources for optimal performance when deploying web server services in each location by implementing a Docker stack that leverages the system's operating system visibility in the swarm detection area. It can be achieved and system performance can be improved. Implementations are done using virtual servers built on virtual machines, support the use of virtualization-level operating systems that leverage the capture system provided by the Docker network, and multiple implementations using the Docker stack implementation. Use the container of Images and web server resources to use directly to another visible server. The virtual server is designed for one node manager and two staff nodes, and the node manager becomes the center for using Docker image creation from the web server, to serve the web in the workplace with given the ability to use Server service provided by the node administrator. This is a collection of servers organized as a network and configured to act as a single entity. Docker directly manages the network between servers in a flock, so you can have a multi-container application with key components on one server, a database on another, and so on. When Docker detects a server crash, it moves the container from the failed server to the default server, whether configured or not, to ensure maximum access to the service. Web applications are spread across multiple content areas and integrate the functionality of all containers. All web front-end related work is done in one container, and all website related work is done in another container. It also uses the Docker network and component features to provide a network between containers and configurations. This can also be achieved by using the Dockerswarm collection to make this structure accessible. Docker Swarm uses a load balancer to measure the load between different containers. Among them, one is considered a large container and the other is considered a slave ship. You can access this web application from your browser or another container. Loading is a method of evenly distributing the load of traffic across two or more lines so that traffic is more efficient, performance is improved, response times are reduced, and communication line congestion is avoided. Load balancing requires immediate failover management. Failover is an automated navigation process that causes downtime in one of the programs and allows the backup process to run on the current system.

IV CONCLUSION

Docker provides the ability to automate containerized applications. It provides a simple and effective workflow and adds an additional layer to the host operating system. Deploying a web application on top of a Docker container is very easy and also provides interoperability and portability. This is a very cost effective and time saving solution in the IT world. Use Docker Networking, Docker Swarm, and Docker Compose. Deploying a layered web application to Docker is very easy. In the Cluster Swarm Service area, the web server is provided by the Docker stack and distributed to the worker nodes registered in the cluster system. Docker swarm plays its role by running a load balancing mechanism to distribute the running load to multiple worker nodes managed by the node manager. The split node is split into two, worker 1 and worker 2. This department is used to perform load balancing to receive high load requests and reduce disasters when the web server service jogging on the worker node is down. Proposed approach show that virtualization at the operating system level supports Docker swarm area failover and load balancing mechanisms, and the Docker stack plays a role in distributing web server services from node managers to node workers as a backup. The server services the web server.

REFERENCES

[1] A. S. Balantimuhe, S. H. Pramono, and H. Suyono. “Dynamic Web Server Cluster Workload Consolidation with Backpropagation Neural Network Approach,” J. EECCIS, Vol 12, No. 2, 2018.

[2] M. R. M. Bella, M. Data, and W. Yahya. “Web Server Load Balancing Based On Memory Utilization Using Docker Swarm,” Proc. 3rd InternationalConference on Sustainable Information Engineering and Technology, pp. 220–223, SIET 2018.

[3] Supramana. “Load Balancing Implementation on a Web Server Using Apache,” J. Information management, Vol. 5, No. 2, 2016.

[4] Y.Alahmad, T. Daradkeh, and A. Agarwal, “Availability-Aware Container Scheduler for Application Services in Cloud,” Conf. 2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC), 1–6. IEEE, 2018.

[5] C. H. Huang, and C. R. Lee, “Enhancing the availability of Docker swarm using checkpoint-and-restore,” Proc. 14th International Symposium on Pervasive Systems, Algorithms and Networks, I-SPAN, 11th International Conference on Frontier of Computer Science and Technology, FCST 2017 and 3rd International Symposium of Creative Computing, pp. 357–362, ISCC 2017.

[6] Y. Li, and Y. Xia. “Auto-scaling web applications in hybrid cloud based on docker”, in Proc. 2016 5th International Conference on Computer Science and Network Technology, ICCSNT 2016, 75–79.

[7] Y. Wu, and H. Chen, “ABP scheduler: Speeding up service spread in a docker swarm,” Proc. 15th IEEE International Symposium on Parallel and Distributed Processing with Applications and 16th IEEE International Conference on Ubiquitous Computing and Communications, ISPA/IUCC, pp. 691–698, 2017.

[8] S. Sharma. “Virtualization & Benefits of Cloud Computing,” https://www.shreysharma.com/virtualization-benefits-of-cloudcomputing/ Apr 26, 2019.