RESEARCH ARTICLE OPEN ACCESS

DOCKER AND ITS FEATURES

Pawandeep Kaur [1], Nitin Pawar [2], Faiz Tanzeel Ansari [3], Rohit Kumar Samad [4], Ghanshyam [5], Gyan Prakash Roy [6]

[1],[2],[3],[4],[5],[6] Department of Computer Science and Engineering
lovely professional university jalandhar, punjab - India

ABSTRACT
Docker offers some facilities useful to developers and managers. It is an open platform that can be used in a compact, lightweight runtime and packaging tool, known as Docker Engine, to create, distribute, and run applications. This also offers Docker Hub, a cloud infrastructure for device sharing. Replacing conventional virtual machine with docker container will reduce costs. It brings down the cost of restoring the cloud development framework excellently.

Keywords **--** Docker, Virtual Machine, Docker Server, Virtualisation, Cloud Computing.

I. INTRODUCTION

Docker is an open-source software that runs programs and making the task of developing and distributing them simpler.The applications built into the docker are bundled into a standard type called a container, with all the supporting dependencies. Such containers usually run in a sandbox environment over the operating system's kernel. In terms of performance, the additional layer of abstraction can have an effect. [1].
Container technology has been around for over ten years, but Docker, a comparatively recent technology, is hoped to be one of the best innovations right now because it incorporates modern elements that previous technologies have not..
Initially it gives the container formation and control facility. Besides that, the developer can easily load applications into lightweight docker containers. This virtualized software can be performed with ease anywhere without alteration. In addition, on the same hardware, the docker can communicate more virtual circumstances than various innovations. Docker can easily communicate with third-party instruments to tie things up, which help to quickly deploy and handle the docker containers. Docker containers can be deployed easily inside the cloud-based environment [2].
This paper is a study of docker technology and will evaluate its results through a systematic analysis of the literature. The article is set out as follows. Next segment introducing the docker technology. A more detailed overview of the docker and its components is provided in Section 3[The following section incorporates Docker technology. Section 3 provides a more comprehensive rundown of the docker and its modules.]. Section 4 briefly contrasts Virtual Machine technology and Docker technology. Sections 5 and 6 tackle, respectively, the advantages and drawbacks of docker containers. In Section 6 and 7 we briefly analyse a few recent works on calculating Docker's efficiency and compare it to other container technologies. Finally, features of virtual machines and containers are outlined briefly of Sections 9 and 10, accompanied by a concise description of this paper.

II. LITERATURE SURVEY

Boettiger, C [1] has presented in his work how Docker Container Technology can be used for operating system virtualization, cross-platform portability, modular re-usable elements, versioning and a ‘DevOps’ philosophy, to address the short coming of simple virtual machines.

B. I. Ismail *et al*. [12] has evaluated Docker as a platform for Edge Computing on 4 fundamental criteria: deployment and termination, resource & service management, fault tolerance, zcaching and have found that docker is a viable Edge Computing platform.

Bui, T. [2 has addressed the security degree of Docker in his work, and the review considers two fields. Docker's inner protection, as well as how it deals with safety features of the Linux kernel, Furthermore, the paper discusses and identifies how to increase Docker Security level while using it.

Wes Felter, et.al [3] explores the performance of traditional virtual machines deployments and contrast them with the use of Linux containers. Their results show that containers result in equal or better performance than VMs in almost all cases.

Service handoff is achieved via container migration. Lele Ma and his fellow researchers proposed a migration method which leverages the layered storage system to reduce file system synchronization overhead. They implemented a prototype system and conducted experiments using real world product applications.

G. Avino quantifies the CPU used by Docker while running two separate multiplayer gaming and video streaming platforms in his article, which explores Docker's suitability in situations involving mobile cloud technology,. The gaming service and the video distribution case provide different outcomes when tested for different numbers of clients and servers.

Chan ho yong uses the vp tree algorithm, which is built on top of the computation approach in the proposed context, to produce successful performance. Although data scales up, the system often follows the system's scalability and fault tolerance criteria. In terms of reaction time, the system over Docker container environment outperforms the VM-based environment.

When using pooled assets, Drew Lucas in his paper provides a strategy for significantly reducing the amount of effort and lead time taken to take autonomy algorithms from initial conception to field trials. The method allows use of the Docker containerization environment, as well as setup and deployment modules that are automated.

Julian M. Bass conducted a series of tests using Docker v1.10 to compare the attack surface of hosts running services within Docker containers to hosts running the same services on Debian Jessie, the base operating system in our article. Using Docker containers enhances the attack surface of a given host, not vice versa, according to our vulnerability analysis.

Cloud vendors are responsible for their services, which must be available, dependable, and portable at all times. In his article, Suchit Dhakate suggests a container-based architecture for automating cloud environment monitoring. A dashboard is being built to display the actual global health status of the cloud, as well as the potential to carry out preventive notifications to deter any unwanted states.

III. DOCKER

Docker offers a container management facility for applications when they are deployed. In a container world, where programmes are virtualized and performed, Docker adds an extra layer of development system on top of it.. The docker is developed to provide a convenient and portable environment in which code can be run easily, as well as an extra feature for the proficient work cycle to take the code from the machine for testing before production [9]. Russell (2015) states that the docker helps you to check and deploy the code into the production environment as quickly as possible [6]. Turnbull (2014) ends by saying dockers are surprisingly simple [9].Of course, a docker with a basic configuration system or a binary docker with a linux kernel may be used to get started.

IV. DOCKER INSIDE

There are four mains internal docker modules, including Docker Client and Server, Docker Images, Docker Registries and Docker Containers. The following parts should describe certain components in detail.

4.1 Docker Client and Server

As shown in Figure 1 Docker can be described as a client and server-based program. The server docker receives the request from the client docker and then processes it accordingly. The full RESTful (Representational State Transfer) API and a binary client command line are shipped via the docker. Docker daemon / server and docker client can run on the same machine or connect a local docker client to a remote server or daemon running on another machine [9].

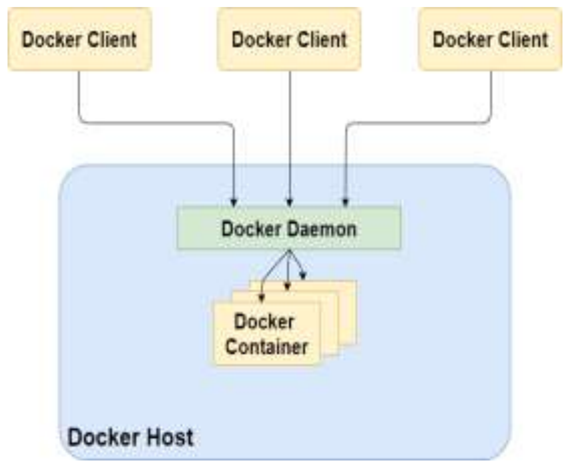

Fig.1 Docker architecture

4.2 Docker Images

There are two ways of creating a picture. The first is the use of a read-only template to create an image. Each picture is based on a base picture. Basically, operating system images are the foundation files, such as Ubuntu 14.04 LTS, or Fedora 20. Operating system images build a container with the ability to fully run OS. One can also create base image from scratch. Additional applications may be applied by changing it to the base image, but a new image needs to be created. The process of creating a new picture is called "engaging in a transition." The second approach is making a docker script. The docker file contains a list of instructions that obey all the instructions provided in the docker file and generate an image when the "Docker build" command is run from the bash terminal. That is an automated way to construct an image.

4.3 Docker Registries

Docker images are put in registries of dockers. This operates correspondingly for collections of source code where images can be pushed or pulled from a single source. There are two types of state and private registries. Docker Hub is also a shared repository where anyone can pull images that are accessible and move their own images without making a scratch image. Using the docker hub feature, images can be distributed to a given area (public or private).

V. VIRTUAL MACHINE VS DOCKER

Virtualization is an old term that was used in cloud computing after IaaS was embraced as a key technique for network constitution, resource supply, and multi-tenancy. Virtualized tools play a key role in solving the issues using cloud computing's main technique. Figure 2 displays the virtual machine's architecture.

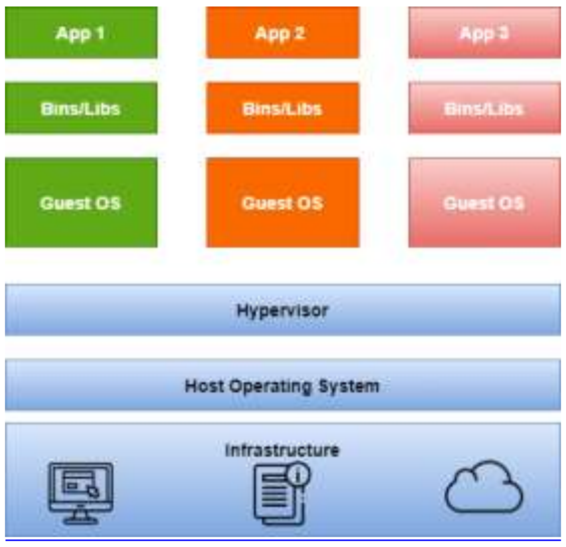

Fig. 2 Virtual Machine architecture

Hypervisor lies between running host and guest systems. It is a virtual network and operates just one server operating system. It operates in between the CPU and the operating system. It is divided into two parts by virtualization: the first is para-virtualization and the second is complete virtualization [3]. Figure 3 portrays the Docker Container design. The docker tool handles the Linux containers and it is used as a framework for virtualizing the operating system level.
Figure 3 indicates that there are several Linux containers in a single control host which are isolated. Linux kernel allocates resources such as Network, Memory, CPU, and Block I / O, and it also deals with groups without starting a virtualizing machine [8].

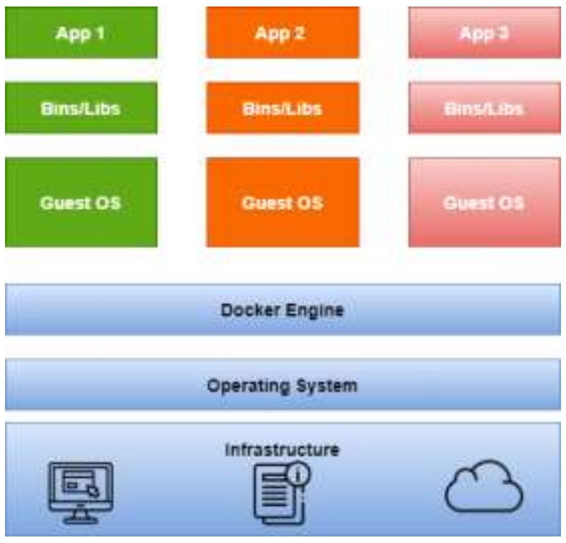

Fig. 3 Docker Container architecture

A Docker Container architecture is to handle the CPU in the Linux containers and spread its power more skilfully. For any case of Hyper-V or VMWare it is not easy to run more than ten virtual machines because of the overhead incurred [13]. The containers have addressed this problem to a great degree. Containers use only certain resources needed for the facilities or applications. So, it is possible to execute more than 50 container requests on a poorly configured system.
Suppose an organization, for example, offers email security services. Such programs have the main functions of monitoring emails for viruses, spam, and malware. Additionally, if the software is deployed in the cloud [10], it will manage to pass messages to the agent, logs, and report delivery failure. Mostly, no related dependencies or OS level libraries or other kernel data structure are used in these situations. Hence, containerizing each part is worthwhile by sandboxing them using OpenVZ or Docker instead of making virtual machines.
Digital computers are used in many businesses to perform element testing. Lots of CPU resources and memory space are used in this process. Whereas container technology offers its users a guarantee that the excess of a workload will not impact the resource output.
Compared to virtual machines, the container takes less time for deployment and container adaptability is much higher than VMs. In fact, both Docker and OpenVZ were under significant scrutiny in terms of their safety aspects. As isolation is decreased it affects the protection directly, which often rapidly decreases. Linux root users can easily access containers, as containers use the same kernel and operating system as well. Docker isolation isn't as powerful as a virtual machine, since the docker isolates the code that runs from its primary host in the docker container.

Furthermore, some of the programs may not be able to run in a containerized environment, so they may need to operate on another operating system.

VI. ADVANTAGES OF DOCKER CONTAINER

Linux container demand and advance can be seen in the last few years. Docker quickly became popular due to the advantages of the docker container. The docker's key advantages are speed, portability, scalability, fast delivery and density.

6.1 Speed

Speed is one of Containers ' most exceptionally assumed advantages. While highlighting the advantages of using a docker, it would be amazing not to consider the docker's pace in the discussion. The time it takes to construct a container is simple, because it's very small. Production, testing and delivery can be done faster, as small containers. Once they have been installed, containers can be moved for testing, and then from there into the production environment [12].

6.2 Portability

Some applications within docker containers are highly portable. It is easy to transfer such portable applications as a single feature and the output remains the same [12].

6.3 Scalability

Docker has the potential to distribute it to multiple physical servers, data servers, and cloud platforms. It can run on any Linux system, too. Containers can be quickly transferred from cloud to local host and back to cloud at a fast speed from there. Adjustments can be made easily; the user can simply adjust the scale to match the need [5].

6.4 Rapid Delivery

A Docker Containers configuration is generic, so programmers do not need to burden each other's activities. The administrator is responsible for deploying and maintaining the server with containers, while the programmer is responsible for handling the applications within the docker container. Containers can operate in any environment because they have all the requisite dependencies embedded in the applications and are all tested [12]. Docker offers a stable, consistent, and improved environment so predictable outcomes can be achieved by moving codes between development, testing, and production systems.

6.5 Density

Docker uses more effectively the resources available, as it does not use a hypervisor. Therefore, more containers can be run as opposed to virtual machines on a single host. A Docker Containers ' efficiency is higher due to higher capacity, and no overhead resource wastage [5].

VII. DISADVANTAGES OF DOCKER CONTAINER

There are some docker container drawbacks, which are listed below [1, 5]:

- Docker does not have complete virtualization because it depends on the linux kernel provided by the local host, but it does not run on older computers.
- Older machines would not be able to access Docker. It only works for 64-bit computers on the local network.
- The docker container will provide the full virtualized environment for both Windows and Mac machines. Even though the boot2docker tool fills this void, it should also be tested whether it prevents users from adopting these systems or if the integration and output with the operating system of the host machine is sufficient [4].

- The risk of security problems needs to be evaluated. Building up trustworthy binaries could be made easier for future support by digitally signing docker files.

- A major concern is to test whether the teaching community or science researcher would consider adopting a docker in a significant way.

VIII. SUMMERY

When containerised, Docker automates the applications. The host operating system is augmented with an extra layer of docker software. Docker performance is better than that of virtual machines, as it has no operating system for visitors and fewer overhead resources.

REFERENCES

[1] Boettiger, C. (2015). An introduction to Docker for reproducible research. ACM SIGOPS Operating Systems Review, 49(1), 71-79.

[2] Bui, T. (2015). Analysis of docker security. arXiv preprint arXiv:1501.02967.

[3] Felter, W., Ferreira, A., Rajamony, R., & Rubio, J. (2014). An updated performance comparison of virtual machines and linux containers. technology, 28, 32.

[4] Harji, A. S., Buhr, P. A., & Brecht, T. (2013). Our troubles with Linux Kernel upgrades and why should care. ACM SIGOPS Operating Systems Review, 47(2), 66-72.

[5] Joy, A. M. (2015). Performance Comparison between Linux containers and
virtual machines. Paper presented at the Computer Engineering and Applications (ICACEA), 2015 International Conference on Advances in.

[6] Russell, B. (2015). Passive Benchmarking with docker LXC, KVM & OpenStack.

[7] Scheepers, M. J. (2014). Virtualization and containerization of application infrastructure: A comparison.

[8] Seo, K.-T., Hwang, H.-S., Moon, I.-Y., Kwon, O.-Y., & Kim, B.-J. (2014). Performance Comparison Analysis of Linux Container and Virtual Machine for Building Cloud.

[9] Turnbull, J. (2014). The Docker Book: Containerization is the new virtualization.

[10] Van der Aalst, W., Weijters, T., & Maruster, L. (2004). Workflow mining: Discovering process models from event logs. Knowledge and Data Engineering, IEEE Transactions on, 16(9), 1128-1142.

[11] Varghese, B., Subba, L. T., Thai, L., & Barker, A. (2016). Container-Based Cloud Virtual Machine Benchmarking. arXiv preprint arXiv:1601.0387

[12] B. I. Ismail et al., "Evaluation of Docker as
Edge computing platform," 2015 IEEE Conference on Open Systems (ICOS), Melaka, Malaysia, 2015, pp. 130-135, doi: 10.1109/ICOS.2015.7377291.

[13] Avino, G., Malinverno, M., Malandrino, F., Casetti, C., & Chiasserini, C. F. (2017, August). Characterizing docker overhead in mobile edge computing scenarios. In *Proceedings of the Workshop on Hot Topics in Container Networking and Networked Systems* (pp. 30-35).

[14] Nguyen, D. T., Yong, C. H., Pham, X. Q., Nguyen, H. Q., Loan, T. T. K., & Huh, E. N. (2016, January). An index scheme for similarity search on cloud computing using mapreduce over docker container. In *Proceedings of the 10th International Conference on ubiquitous information management and communication* (pp. 1-6).