International Journal of Advanced Information in Engineering Technology (IJAIET)
Vol.5, No.5, May 2018 ISSN: 2454-6933

Deploying of Micro-Services and DevOps Pipelining on Docker

H.Anwar Basha, C.Sabarinathan, Ajay Paul A, Paul X. Lloyd, Hari Prabu I.
Department of Computer Science and Engineering
SRM Institute of Science and Technology,
Chennai, Tamil Nadu, India

Abstract

The Virtualization is an emerging technique which helps in the efficient allocation of resources. Nowadays most of the start-ups prefer renting micro services for their business which can be provided by containers. Recently, containerization has become the cutting edge technology in the virtualization platform. The automation technique implemented in this paper helps in increasing productivity by significantly reducing the time consumed. The scope of the paper is to primarily provide high availability for the micro services deployed in the containers. The use of Docker's Swarm clustering algorithm provides an advantage over the traditional hypervisor by reducing the downtime of the instance in case of a disaster and since it uses Containers (thin provision OS) instead of VMs(Virtual machines) it utilizes resources efficiently by using only library files required for the particular application.

Keywords – Docker; Automation; Hypervisor; Virtualization; Containers; Ansible; Jenkins; Github.

Introduction

Nowadays the automation is one of the trending techniques which most of the companies are moving towards for automating their tasks as it is much more easier and the day-to-day tasks of the organization can be automated to increase productivity. The automation technique provides more efficiency and it is fail proof method than the traditional technique. This paper proposes an automation system where the whole pipeline of the software development lifecycle (SDLC) is being automated and provides high availability to the services deployed on the production with fault tolerance. It is a systematic approach to develop software into production. It creates a structure for the developer to design, create and deliver high quality software according to the requirements of end customer. It also provides efficiency for the deployment of the desired product. The purpose of ***SDLC*** process is to provide a more fast process of production which is cost efficient and of high quality. Traditionally the waterfall model was used it is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the all phases of the production from compiling to deployment. Then came in the Agile model The functional modules are developed simultaneously as prototypes and are integrated to make the complete product for faster product delivery. The customer gets early visibility in the software and can provide feedback on design, delivery,and process is much more faster than Waterfall model but it is slower than the Devops model When evaluated Docker Containers was found to be more efficient than the VMs hence the Docker containers have been used for micro service deployment for the Small and Medium Scale Industries and a clustering technique called as swarm cluster has been used to provide high availability for the containers. This whole container as a service (CAAS) technique has been deployed on a pipeline by using DevOps Automation method by using Continuous Integration tools like Git which is used as source code management and a version controlling system, Jenkins which is used as a code building tool using plug-ins and Continuous Deployment tools like Docker which is the infrastructure on which the Containers are being deployed, Ansible which is used for scripting for integrating the whole pipeline. This method is the most latest and efficient method of delivering a service which brings down the time taken for production from months to hours and delivered to the customer.

Related Work

The fact relates the virtualisation of Operating system provides a common way to run different services in the cloud [8]. In which, VMs plays major roles in the field or domain of cloud computing. However, it plays vital roles, it has drawbacks like resource consumption based on consumption of RAM and time [9]. To overcome those problems, A new technology Docker is used. Where docker is said to be lightweight container. Results show that containers achieve better performance when compared with traditional Virtual Machines [8]. In which Docker sets a new changes in the cloud computing. Docker is mainly developed especially based on performance and resource consumption [1]. Compared with VMs, Docker has more advantage in the terms of data intensive app and computing ability [1]. In which container-based virtualization uses single kernel to multiple instance on OS. Hypervisor of Docker uses a thin kernel called base layer from the VMs machine. The biggest challenge is to access hardware without virtualizing drivers for hardware, in which every VM has its own kernel and hypervisor [2]. Container base

technology is used as an efficient technology for running high performance applications. In this docker is used as container based technology and VMs created using VMware [3]. Comparison of Virtual Machines and Linux Containers which it explores the performance of traditional VM deployment and contrast them with the use of Linux containers. We use KVM as hypervisors of Docker as container manager. This work proposed Docker Container based Virtualization which has recently emerged as alternate light weight technology. In which it is an open source engine that automates the deployment of applications into Container [7]. Docker Swarm mode container technology will make the task of deploying multiple machines easily with less resource allocation [6]. This new deployment technology can reduce the effort and cost for deploying a micro service based application with minimum downtime and helps in achieving higher scalability [4]. In which compared with VMs Docker has high compatibility during Up and Down time, and has efficiency while automation process.

Proposed Design and Methodology

The design proposes an end to end Automation of the DevOps pipeline and delivering Container as a service (Caas) with high availability. The proposed design consist of three main process Virtualization, Containerization and Automation by using VMware, Docker, Git, Jenkins, Ansible tools. The system consist of physical server where the VMware is used to virtualize the hardware above which a Linux operating system is deployed, the type 2 hypervisor is used in this process. Above the Linux machine the Docker engines are employed above which the Container is used to provide Microservices.The Docker containers are used instead of VMs since the Container is a thin provisioning OS which consist of only the library and binary files required to run the application used in the particular container, it does not require a Kernel to run the application instead is uses the Kernel of the underlying linux machines, the Docker also uses the swarm clustering algorithm to provide high availability for the services provided.

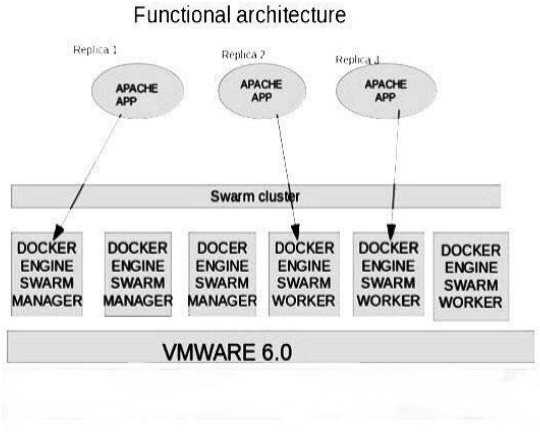


Figure1

The Docker Swarm Clustering is a clustering model which is used to group a set of nodes which are used to provide the Micro service. The swarm cluster model consists of 3 manager nodes minimum and specific number of worker nodes based on the purpose which can also be scaled up automatically, the manager nodes controls and manages the worker nodes, the worker node is used to deploy application and controls the Containers. For each service specified number of replicas are created in the worker nodes thus in case of a failure the replicas will be active, in case of VMs the process has to be done manually and there is downtime associated with it but through the Docker swarm this has been overcome. This whole Docker setup will be deployed by using DevOps model where the whole pipeline of SDLC will be automated, consider a HTTP service to be hosted, as soon as the code is added in the GIT repository the code is evaluated and compiled by using the Jenkins by using Maven and an Ansible script is used to create an infrastructure in the Docker environment where the Swarm Clustering method has been used, hence the service delivered is fast by Automation technique, efficient by using DevOps model and also fail proof by using Docker Swarm, therefore providing service more efficiently.

Component	Tools used
Virtualisation	Vmware
Containerisation	Docker
Source Repository	GIT Repository
Build Server	Jenkins
Container Repository	Docker Hub
Scripting	Ansible

Table1

Figure 3

Experimental Setup and Evaluation

Deploying the Distributed applications above the Docker Containers we found so many differences during our practical experiments. The Configuration of each virtual machines are 1 VCPU and 20GB of disk and 1GB of RAM and CentOS 7.4.Before deploying we did so many experiments to evaluate the main differences and the efficiency of running a microservice on a container when compared with the Virtual Machines. In this we evaluated the Virtual Machines design and the Container design and some other things like Deployment time and response time, efficiency, pipelining model etc.. The experiments of the application was evaluated and the results are discussed below.

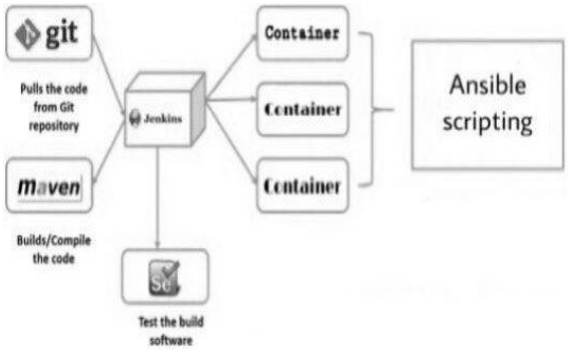
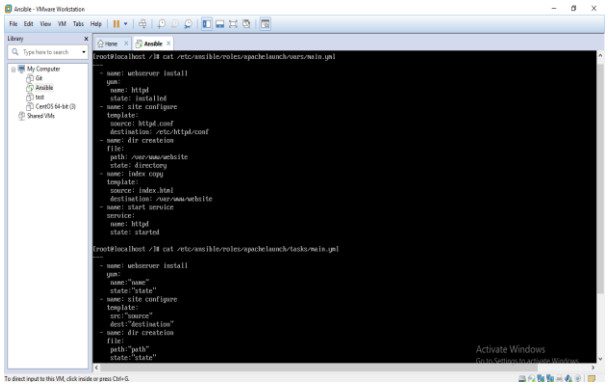


Figure 2

Traditional Configuration Vs Automation

Automation is the most emerging technique in the IT sector. Ansible is used for end-to-end Automation of the SDLC which increase productivity and efficiency by reducing time taken. The Ansible configuration consist of specific roles for each service and specific tasks and vars directory for each service the parameters required for each services will be added in the vars file and tasks file acts as a template file for the service, any changes in the parameter of the service can be made in the vars folder itself without accessing the tasks file, this increases efficiency.



Virtual Machines Vs Docker Containers

Since microservices are independent of each other, virtualized environment is possible which makes the system more secure and easy to maintain. In virtual machines the virtualization happens in the hardware level it uses the whole operating system to deploy the application hence resource allocation is high. Due to the use of operating systems there compatibility issues for application while migration in VMs

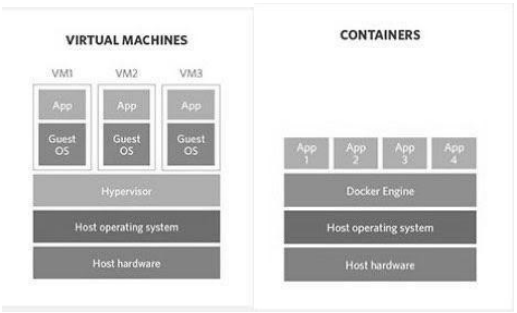


Figure 4

Containers are easy to manage due to their lightweight property and they do not have any kernel but they have the binary and library files required to run the application . In containerisation the virtualization happen in the OS level. As containers uses isolated process to deploy application the resource allocation is very less We also compared the deployment time for both VM and the docker containers for the web hosting service in which containers are taking considerably less time to deploy than Virtual Machines since VMs needed to build the operating system first then the application but the containers directly deploys the application. There is no compatability issues in the container as there is no operating system.

	Centos VMs	Centos Container
Size	1085 MB	325 MB
Deployment Time	20 Mins	5 Sec
Downtime	15 Sec	NIL

Table 2

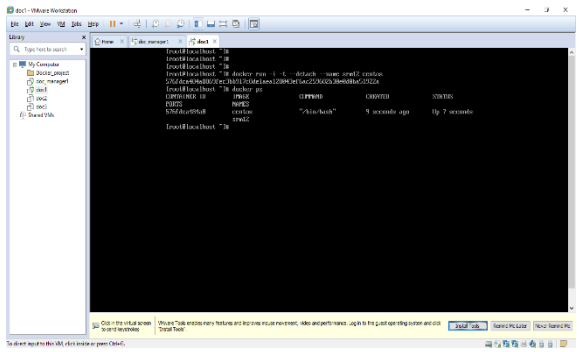


Figure 5

Docker Swarm Vs Kubernetes

The platform of Docker is an open-source. The Container technology is discussed, whenever docker is mentioned Docker Swarm is only meant. Docker is a container storage platform which is flexible and very easy of use. Docker Swarm, is all about management of a cluster of engines. Another open-source container storage program which is popular is the Kubernetes. It was built manage google own systems by themselves. Kubernetes is an useful open-source, powerful and scalable tool. It containers can be handled and offers scalability at immense level and automation simultaneously. but, it is very hard to use and many developers often avoid working in it. Installation and setup process differ to each tool. Set up in the Cloud or other infrastructure to Manage the containers always depends a lot on how it is works. while comparing, Kubernetes is tough. Installation and setup of these systems, it is a hard time to the developers. Primarily for every operating system(OS) it needs to be reconfigured. Although the online documentation helps a lot in the process, while building a custom environment things can get complex. To use Google is your only solution to your advantage. The planning that is required before you start implementing that is why the setup and installation becomes a lot more difficult. Planning of the nodes needs to be done which can take a considerable amount of time and effort, manual integrations are also needed which adds up to the burden. Automation cannot be done, and hence it is hard to manage the kubernetes. Docker, hence it is a Command Line Interface (CLI), setup and management becomes much more easier. The CLI and GIT-like semantics are being used by Docker Swarm. This gives application developers ease to incorporate their workflow new technologies. Compared to Kubernetes, nothing new to be learned while the container is being implemented for new operating system or environment. Overall, when it comes to setup and installation Docker wins. Scalability they offer is the top notch for using container services. Both platforms are highly scalable and support thousands of containers at a particular time. Initially, Docker had a great support for a less number of containers only. With the series of updates it can support as many containers when compared to Kubernetes. 1000 node clusters can be suppported by two systems. The 100 nodes can manage cluster 3000 containers. Kubernetes is far better to Docker in performance. However, the research suggests that Docker could work containers five times faster when compared to Kubernetes. In spite of the hard installation process Kubernetes is widely used by the large of developers, the flexibility it offers and backup by Google, which is one of the leading companies accounts for its popularity. Docker is a small community. The large community of Kubernetes can adopt new tools, support and features. Small developer needs to learn container services can use Docker to start with. Hence it is better to use Docker rather than Kubernetes.

Advantages of Docker Containers

Rapid deployment containers include the minimal runtime requirements of the application, as they have no Kernel of their own thus reducing their size and deploying them more quicker. Portable across platforms an application its dependency and libraries can be grouped into a single container that is independent of Linux machine kernel, platform distribution, or deployment model. This container can be migrated to another system that runs on docker, and can be executed there without any compatibility problems. Version controlling and reusability you can track chronicle versions of a container, investigate their difference, or roll-back to older versions. Containers can reuse things from the previous layers, which makes them thin. Shared resource you can make use of remote repository or private repository to share your containers. Red Hat registry can be used for this purpose. Thin provisioning Docker images are usually very small, which comforts rapid deployment as well as delivery thus reducing the time for launching new application containers. Ease of maintenanc Effort and risk of problems for application dependencies are minimal in Docker.

Results

From the proposed method of deploying services in a docker container over an Devops pipeline, a faster production practice with the high availability and integrated load balancing of the service are delivered which increases the efficiency of both cost and time for running a microservice.

Conclusion and Future Work

Hence from the above evaluated experiments the Docker Containerization proves to be an efficient way than VMs to provide the service to the customer by providing high availability, utilizing less resource and lesser time consumption. Secondly the automation of the whole pipeline of the software development lifecycle has increased efficiency manifold providing benefits of cost and time to the organization and a swift delivery of service to the customer hence proving useful to both the parties. The future enhancement would be an effective way to reduce the compilation and the infrastructure building time taken by the automation system and a method used to provide service to the organization by deploying private cloud using the technology Openshift launched by Redhat which provides isolation to the service provided by deploying Openstack over the Docker.

Acknowledgement

We would like to show our gratitude to Mr.C.Sabarinathan, , SRM University and Mr.H.Anwar Basha, Asst. Prof, SRM University for sharing their pearls of wisdom with us during the course of this research, and we thank reviewers Mr.Rajive Gandhi, Mr. Ethirajulu, Mr.Arun Nehru for their so-called insights, although any errors are our own and should not tarnish the reputations of these esteemed persons.

References

1. Wes Felter, Alexander Ferreira, Ram Rajamony, Juan Rubio, “An updated performance comparison of Virtual Machines and Linux Containers”.
2. Roberto Morabito, Jimmy Kjallman, and Miika Komu, “Hypervisors vs. Lightweight Virtualization: A Performance Comparison”.
3. Nitin Naik, “Building a Virtual System of Sytems using Docker Swarm in Multiple Clouds”.
4. Nikyle Nguyen, Doina Bein, “Distributed MPI Cluster with Docker Swarm mode”.
5. Vindeep Singh, Sateesh K Peddoju, “Cointainer based Microservice Architecture for Cloud Applications”.
6. Fawaz Paraiso, Stephanie Challita, Yahya Al-Dhuraibi, Phillippe Merle, “Model Driven Management of Docker Containers”.
7. Theodora Adufu, Jieun Choi, Yoonhee Kim, “Is Container-based Technology a winner for high performance Scientific Applications?”.
8. Akos Kovacs, “Comparison of Different Linux Containers”.
9. Sachchidanand Singh, Nirmala Singh, “Containers and Docker: Emerging Roles and Future of Cloud Technology”.
10. https://www.docker.com/engine/swarm
11. https://www.docker.com