Software virtualization using containers in Google cloud platform

Sirisha Potluri, Konda Varshith

Abstract: ***Virtual machines have been the main source of deployment of production ready applications since the beginning of cloud computing, but virtualization has its negative effects on resource management, due to this operating system-level virtualization called containers are gaining popularity in recent times. Software Visualization in Cloud Computing allows the single computer server to run one or more virtual environments. It is quite similar to virtualizations but here it abstracts the software installation procedure and creates a virtual software out of it. In software virtualizations, an application will be installed which will perform the further task. One of the software is physical while others are virtual as it allows 2 or more operating system using only one computer. Docker is used to separate applications from infrastructure so we can deliver software quickly. With Docker, we can manage your infrastructure in the same ways you manage your applications. Software virtualization can be achieved through this platform. Using Docker containers software virtualization is made simple and results are provided in this paper.***

Keywords: Cloud computing, Virtualization.

I. INTRODUCTION

Software Visualization in Cloud Computing allows the single computer server to run one or more virtual environments. It is quite similar to virtualizations but here it abstracts the software installation procedure and creates a virtual software out of it [1-3]. In software virtualizations, an application will be installed which will perform the further task. One of the software is physical while others are virtual as it allows 2 or more operating system using only one computer [4-6]. With Docker, we can manage your infrastructure in the same ways you manage your applications.

II. THEORY

There are many advantages of software virtualization in cloud computing [7-10].

A. Testing: After the testing, the VM can move or delete for the further testing.

A. Virtual machines in Cloud: GCP's unmanaged compute service is Google Compute Engine. You can think of Compute Engine as providing an infrastructure as a service (IaaS), because the system provides a robust computing infrastructure, but you must choose and configure the platform components that you want to use.

Revised Manuscript Received on May 06, 2019

Sirisha Potluri, Department of Computer Science and Engineering, Faculty of Science and technology, ICFAI Foundation for Higher Education, Hyderabad

Konda Varshith, Department of Computer Science and Engineering, Faculty of Science and technology, ICFAI Foundation for Higher Education, Hyderabad

B. Utilization: In software virtualization, there is higher efficiency in resource utilization if it tunes correctly.

C. Efficiency: It is efficient in a way such that it can run 12 virtual machines and eliminates the use of 12 physical boxes. This is the power cost as well as the cost of maintaining the server.

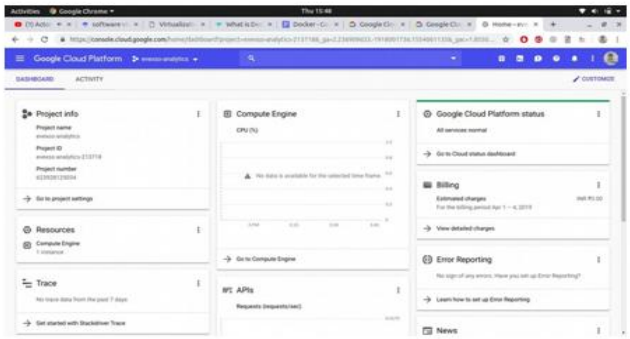
D. Less Downtime: The software is upgrading and the upgrade in the VMs can do when the VM is working.

E. Flexibility: It provides flexibility to the user so that the user can modify the software as per their demand.

III. RESULTS

Every new Google cloud platform project has: project name, project ID, and a project number, which GCP provides. In this example: DCC-cloud-project is the project name. DCC-cloud-213718is the project ID. 623928125034 is the project number as shown in the result page 1. Each project ID is unique across GCP.

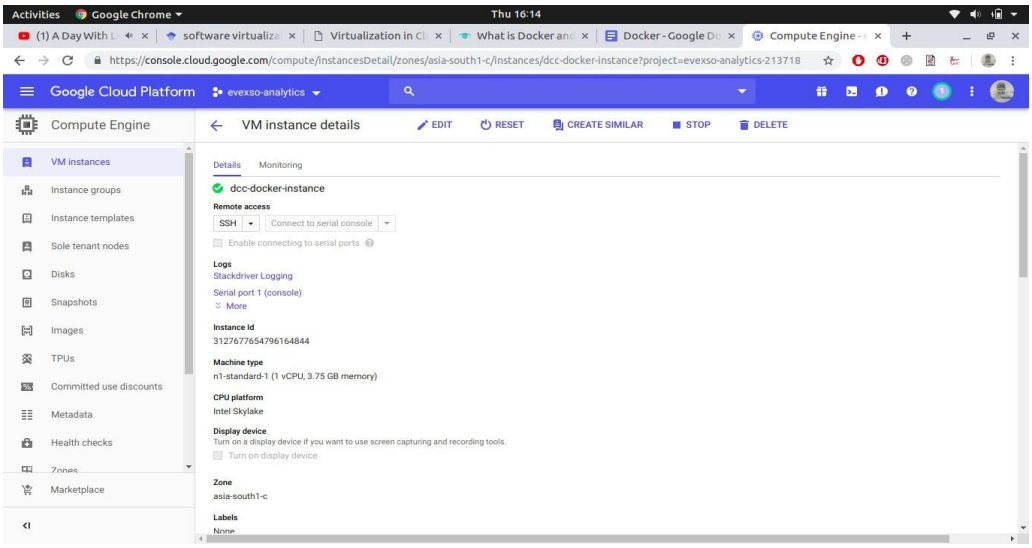
Result page- 1 (Google cloud platform project name space)



B. Google ensures all the factors like availability, reliability, and ready for you to use features.

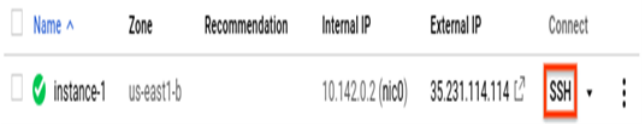
Connecting to instances: The result page 2, describes some of the most common ways to connect to your Compute Engine Linux instances and Compute Engine Windows instances. For additional ways to connect to instances, we can use Connecting using third-party tools, including PuTTy and Connecting to instances that have no external IP addresses. Access your instance before you are connecting to the system.

This page assumes you've followed theQuickstart using a Linux VM guide or the Quickstart using a Windows VM guide to create your instance, which includes creating default user access. Complete at least one of those guides before continuing.



Result page- 2 (Most common ways to connect to computing instances)

C. Linux instances connection: Linux instances can be connected through the Google Cloud Platform Console is as follows:



Result page- 3 (connection to Linux instance)

D. Responsive deployment and scaling: It runs more workloads on the same hardware Docker is lightweight and fast. It provides a viable, cost-effective alternative to hypervisor-based virtual machines, so you can use more of your compute capacity to achieve your business goals.

E. Difference between image and container: An image is a read-only template with instructions for creating a Docker container.

The following Docker command runs an ubuntu container

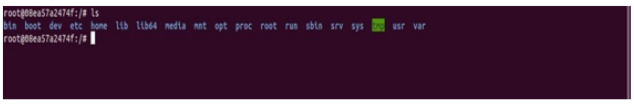
$ docker run -i -t ubuntu /bin/bash

F. Installing and running Ubuntu image inside a container: Command for pulling Ubuntu image from Docker registry as shown in result page 4.

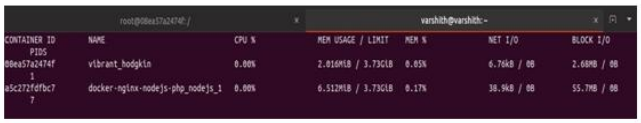
Result page-4 (Installing and running Ubuntu image inside a container)

Command to run Ubuntu container terminal - >$docker run -it ubuntu bash

Checking system files in home directory



Checking the status of each container

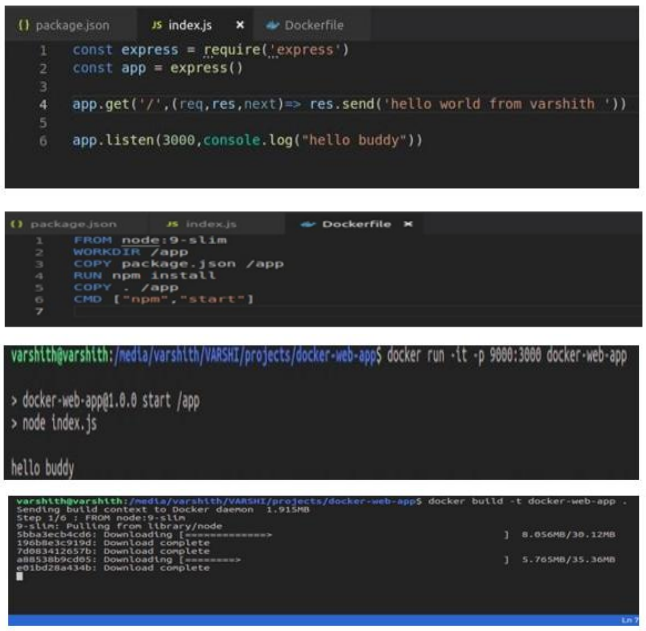


Using Docker stats command we can see current state of the containers regarding memory usage, Cpu usage, network I/O, container ID.

Docker file doesn't have file extensions it is specified as "Dockerfile" Here we will run a NODE.js application in a container as shown in the following result page -5.

We are composing a docker file to run node application in container. FROM indicates the base image of the container. WORKDIR establish the app directory COPY copies package file to app directory RUN will be used to install package.json. CMD will run specified command in container terminal

Result page-5(simple Node js server)



G. Starting the Application with Docker Containers:

Install Docker for Windows or Docker for Mac (If you're on Windows 7 install Docker Toolbox: http://docker.com/toolbox).

Open a command prompt. Run the commands listed in node.dockerfile (see the comments at the top of the file).

Navigate to http://localhost:3000. We need change the IP address if it is asfigned with VirtualBox.

Starting the Application with Docker Compose.

Install Docker for Windows or Docker for Mac (If you're on Windows 7 install Docker Toolbox: http://docker.com/toolbox).

Open a command prompt at the root of the application's folder.

Run docker-compose build.Run. Run docker ps -a and note the ID of the Node container.

Run docker exec -it <nodeContainerID> sh (replace with the proper ID) to sh into the container

Run node dbSeeder.js to seed the MongoDB database

Type exit to leave the sh session

IV. DISCUSSIONS

Virtual machines have been the main source of deployment of production ready applications since the beginning of cloud computing, but virtualization has its negative effects on resource management, due to this operating system-level virtualization called containers are gaining popularity in recent times. Software Visualization in Cloud Computing allows the single computer server to run one or more virtual environments. It is quite similar to virtualizations but here it abstracts the software installation procedure and creates a virtual software out of it. In software virtualizations, an application will be installed which will perform the further task. One of the software is physical while others are virtual as it allows 2 or more operating system using only one computer. Docker is used to separate applications from infrastructure so we can deliver software quickly. With Docker, we can manage your infrastructure in the same ways you manage your applications. Software virtualization can be achieved through this platform. Using Docker containers software virtualization is made simple and results are provided in this paper.

References

1. Kyoung-Taek Seo, Hyun-Seo Hwang, Il-Young Moon, Oh-Young Kwon, Byeong-Jun Kim, Performance Comparison Analysis of Linux Container and Virtual Machine for Building Cloud, Advanced Science and Technology Letters Vol.66 (Networking and Communication 2014), pp.105-111.
2. David Bernstein, Containers and Cloud: From LXC to Docker to Kubernetes, IEEE Cloud Computing, Volume: 1 , Issue: 3 , Sept. 2014, Page(s): 81 – 84.
3. Bhaskar Prasad Rimal, Eunmi Choi, Ian Lumb, A Taxonomy and Survey of Cloud Computing Systems, 2009
4. ZhanibekKozhirbayev, Richard O.Sinnott, A performance comparison of container-based technologies for the Cloud, Future Generation Computer Systems, Volume 68, March 2017, Pages 175-182.
5. Muhamad Fitra Kacamarga, Bens Pardamean, Hari Wijaya, Lightweight Virtualization in Cloud Computing for Research, International Conference on Soft Computing, Intelligence Systems, and Information Technology, 2015.
6. Davide Mulfari, Maria Fazio, Antonio Celesti, Massimo Villari, Antonio Puliafito, Design of an IoT Cloud System for Container Virtualization on Smart Objects, European Conference on Service-Oriented and Cloud Computing, 2015.
7. Fazio, M., Celesti, A., Villari, M.: Design of a message-oriented middleware for cooperating clouds. In: Canal, C., Villari, M. (eds.) ESOCC 2013. CCIS, vol. 393, pp. 25–36. Springer, Heidelberg (2013)
8. Xavier, M., Neves, M., Rossi, F., Ferreto, T., Lange, T., De Rose, C.: Performance evaluation of container-based virtualization for high performance computing environments. In: 2013 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), pp. 233–240 (2013)
9. James Turnbull James Turnbull, The Docker Book: Containerization Is the New Virtualization, 2015.
10. Jeff Nickoloff, Docker in Action, Manning Publications Co. Greenwich, 2016.

Manuscript details:

Sirisha Potluri (Corresponding Author)
Department of Computer Science and Engineering,
Faculty of Science and technology, ICFAI Foundation for Higher Education,
Hyderabad-501203.
sirisha.potluri@ifheindia.org

Konda Varshith
Department of Computer Science and Engineering,
Faculty of Science and technology, ICFAI Foundation for Higher Education,
Hyderabad-501203.
kvarshith@outlook.com