Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Emerging & Future Trends

Name : Haoyi Sun

Student Number : 1904101

Module: Emerging and Future Trends

02/02/2020

1. Introduction

What is cloud computing? Before the computer was invented, there was no network, and each computer (PC) was a single machine. And application software to complete their work. Later, with the Network, single machines can exchange information and work together. Later, the performance of the single machine became stronger and stronger, and there was a server. People have found that some servers can be put together and placed in the computer room, and then users can access and use computer resources in the computer room through the network. Then again, small networks became large networks, and there was the Internet. A small computer room becomes a large computer room, and there is an IDC (Internet Data Centre). When more and more computer resources and application services (Applications, such as watching webpages, downloading movies) are centralized, it becomes "Cloud Computing". Countless large computer rooms have become the "cloud". [1]

Cloud computing definition: Cloud computing is a mode that can be easily accessed on demand based on a network-based, configurable shared computing resource pool. These configurable shared resource computing pools include networks, servers, storage, applications, and services. And these resource pools can be quickly provided and released with minimal management or through interaction with service providers.

2. Emerging and Future Trends Questions1-3

2.1 Question 1

In cloud computing we often use the terms Platform as a Service (PaaS) and Container as a Service (CaaS) when referring to the deployment of services. Compare and contrast these approaches and provide arguments for the role that Docker plays in providing such services. In your answer, identify the benefits and drawbacks of Docker.

The principle of cloud computing is simple. To put it plainly, it is to put computer resources together and put them on the network. However, the implementation of cloud computing is complicated. For example, in the "Double 11" Taobao shopping festival in China, billions of users around the world visit Alibaba's website, and the super heavy traffic requires the design of a cloud with super large capacity, super high simultaneous access, super-fast speed, and super security. The computing system can meet the

requirements of stable business operation. This is the characteristic that modern cloud computing must have. [2]

Cloud computing resources are divided into three levels, which are Iaas, Paas, and Saas
IaaS: Infrastructure-as-a-Service;
SaaS: Software-as-a-Service;
PaaS: Platform-as-a-Service.
The first level is the lowest level of hardware resources, including CPU (computing resources), hard disk (storage resources) and so on. The second level, to some advanced, from the hardware, the need for software applications, such as operating system (Windows, Linux, etc.), database software. The third level, more advanced, you not only have to install the basic operating system, but also the specific service application software, such as FTP server software, online video server software, I can directly use the service.

Different users sometimes just want to run their own simple programs. In order not to affect each other or be affected by the system configuration environment, a virtual machine must be established. If you build a whole set of virtual machines in order to run a small program, it is obviously a waste of resources, and the operation is more complicated, and it will take a long time. And, sometimes, if you want to migrate your own service program, you need to migrate the entire virtual machine. Obviously, the migration process can also be complicated. Is there a way to be more flexible and faster? Yes, this introduces the "Container".

Docker
Containers are also virtualized, but belong to "lightweight" virtualization. Its purpose is the same as the virtual machine, to create an "isolated environment”. However, it is very different from the virtual machine-the virtual machine is an operating system level resource isolation, and the container is essentially a process level resource isolation. And Docker, which is often heard, is a tool for creating containers, and it is an application container engine. Docker's logo is a whale carrying many container boxes. [3]

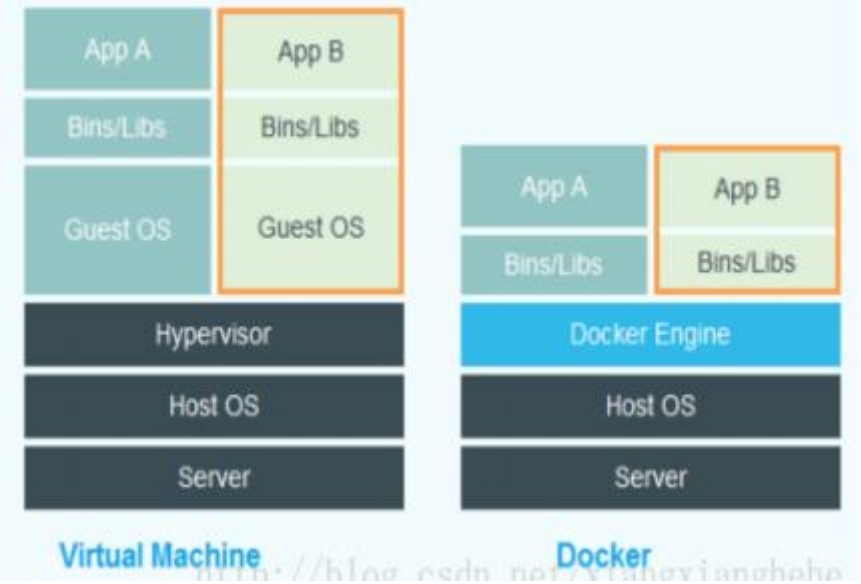

Figure 1 Virtual Machine & Docker

The three core concepts of Docker are a warehouse, an image, and a container. There are images in the warehouse, and the container runs on top of the image.

A repository is similar to a code repository, and is a place where Docker stores image files centrally. Divided into two forms of public warehouse and private warehouse. The largest public repository is the DockerHub mentioned earlier. Of course, private repositories can also be built in the LAN.

An image is similar to a virtual machine image. It can be understood as a read-only template for Docker and contains a file system. Images are the basis for creating Docker. All docker containers run on images.

Containers are the aforementioned containers. They are application instances created from images. They can be started, started, stopped, and deleted. These containers are isolated and invisible to each other.

The image corresponds to a public class written in Java code, and the container is an instance that runs based on this code. Everyone can call this public class, and can pass different parameters according to their needs to form instances of different content, such as We can specify the instance name by passing parameters to start the basic weblogic image container. On the same machine, we can start many processes with different instance names based on the same image. But these parameters do not change the original code, all changes are only stored in the container. So the image is read-only and the container is writable.

Docker is a container. Containers standardize freight goals, and Docker standardizes applications. Containers contain any type of app, each running in its own container, isolated from each other, and sharing the resources of the Great Shark Freighter. This packaged container can be placed in to run on the platform. Very graphically shows the characteristics of Docker: Build, Ship and Run Any App, Anywhere! Run any application on any platform! Compared with traditional virtual machines, the advantages of Docker are obvious. It has a fast start-up time, seconds, and high utilization of resources (a host can run thousands of Docker containers at the same time). In addition, it takes up very little space. Virtual machines generally require several GB to tens of GB, while containers only need MB or even KB.

Benefits and Disadvantages of Docker	
Benefits	
1.Quick configuration	Provides an independent development environment for easy development. Since docker can be used in a variety of environments, and the infrastructure is not required to be associated with the environment of the application, there are no compatibility issues.
2. Improve work efficiency	This is a concept similar to a container. At the beginning of development, the project was placed in an independent environment similar to a container for development. When it was deployed, the container environment was only deployed, and the container environment has been assigned a unified standard by all systems , There are no compatibility issues, so work efficiency can be improved.
3.Application isolation	Docker provides a container for running applications in an isolated environment. Each container is independent and allows the execution of any type of application, without the problems of intrusion and virus infection, and can even use sandbox for virus isolation.
4. Gathering	It is a clustering and scheduling tool for docker containers. Swarm uses the Docker API as its front end, which helps us

	use various tools to control it. It can also control the docker host cluster as a virtual host. This is an ad hoc engine group used to enable a pluggable backend.
5. Routing network	It is able to route incoming requests for published ports on the available nodes to the active container for connection.
6. Services	A service is a task list that allows specifying the status of containers within a cluster. Each task represents an instance of a container that should be running, and Swarm schedules them between nodes.
7.Safety Management	Security and management should be a high-priority consideration; business users should no longer consider them as the last step in migrating applications to containers. Conversely, companies must plan security and management right from the start, incorporate their functions into the application development process, and actively focus on these aspects while the application is running.
Disadvantages	
1. limitation	Docker is based on Linux64-bit and cannot be used in 32-bit Linux / Windows / unix environment
2. Virtualization schemes such as isolation index KVM are still somewhat lacking, and the runtime of common parts of all containers	
3. Implementing Docker is complicated. It requires a lot of technical support, such as container management, orchestration, application packaging, network between containers, data snapshots, and so on.	
…[4]	

2.2 Question 2

There is a trend amongst cloud computing users to move towards developing infrastructure as code (IaC). Explain this term then critically evaluate the future usage of this technology.

The rapid advancement and evolution of technology also requires frequent changes in the configuration of infrastructure. In this process of rapid change, infrastructure needs to be both flexible, secure, and reliable.

Traditional infrastructure operation and maintenance management has the following problems.

---The product team uses the application system to obtain server resources. There are multiple approval processes in the middle, and it needs to wait for the implementation of the operation and maintenance team. The response is not timely.

---Lack of cascading in automation. Although there is a certain degree of automation, it cannot be unattended and requires some temporary command intervention. Due to the high cost of environmental release and reconstruction, it cannot be released, resulting in low resource consumption.

----Disconnect from the product team. It is difficult to dynamically increase the environment at any time based on demand. Additional documentation is needed to describe the environment and may not be updated in time.

In response to the above issues, infrastructure as code is a way to use new technologies to build and manage dynamic infrastructure. It regards infrastructure, tools and services and the management of infrastructure as a software system, and adopts software engineering practices to manage changes to the system in a structured and secure manner. [5]

There are four key principles of infrastructure as code.

--- Regenerative. Any element in the environment can be easily copied.

---Consistency. The configuration of each element of the created environment is exactly the same at all times.

--- Quick feedback. Ability to make changes frequently and easily, and quickly know if the changes are correct.

--- Visibility. All changes to the environment should be easy to understand, auditable, and version controlled.

The goals of infrastructure as code are:

---standardization. Define the environment with code, and standardize the development environment, test environment, and production environment.

---automation. Drive your code preparation environment with automation tools. This includes creating environments, updating environments, and destroying environments.

--- Visualization. Use monitoring to visualize environmental information. The current

state of the environment is visible, and the history of the environment change is visible and traceable.

In addition, if you want to create scalable services in a production environment, you also need to rely on the practice of cabin facilities as code. During peak times, the system can automatically create and join new nodes to achieve dynamic expansion based on the defined environment, and destroy them during low peaks. When a node is found to fail during monitoring, the system can automatically create a new node to replace the failed node according to the defined environment to achieve automatic disaster recovery.

Infrastructure as code is a technology that conforms to the development of the times. After that, programmers will save a lot of time and program based on modularity. The consequential disadvantages may be lack of basic innovation and system rigidity.

2.3 Question 3

Kubernetes is a technology used for the orchestration (management) of Docker containers. Investigate this technology and provide a scenario that best illustrates its purpose.

Kubernetes——Container Orchestration

Kubernetes is a container cluster management system. Its main duties are container orchestration, including starting containers, automating deployment, expanding and managing container applications, and recycling containers. In simple terms, Kubernetes is a bit like a container nanny. It is responsible for managing which machine the container is running on, monitoring the container for problems, controlling the container's communication with the outside world, and more. Through the following K8S system structure diagram, we can see the relationship between Kubernetes and the container. [6]

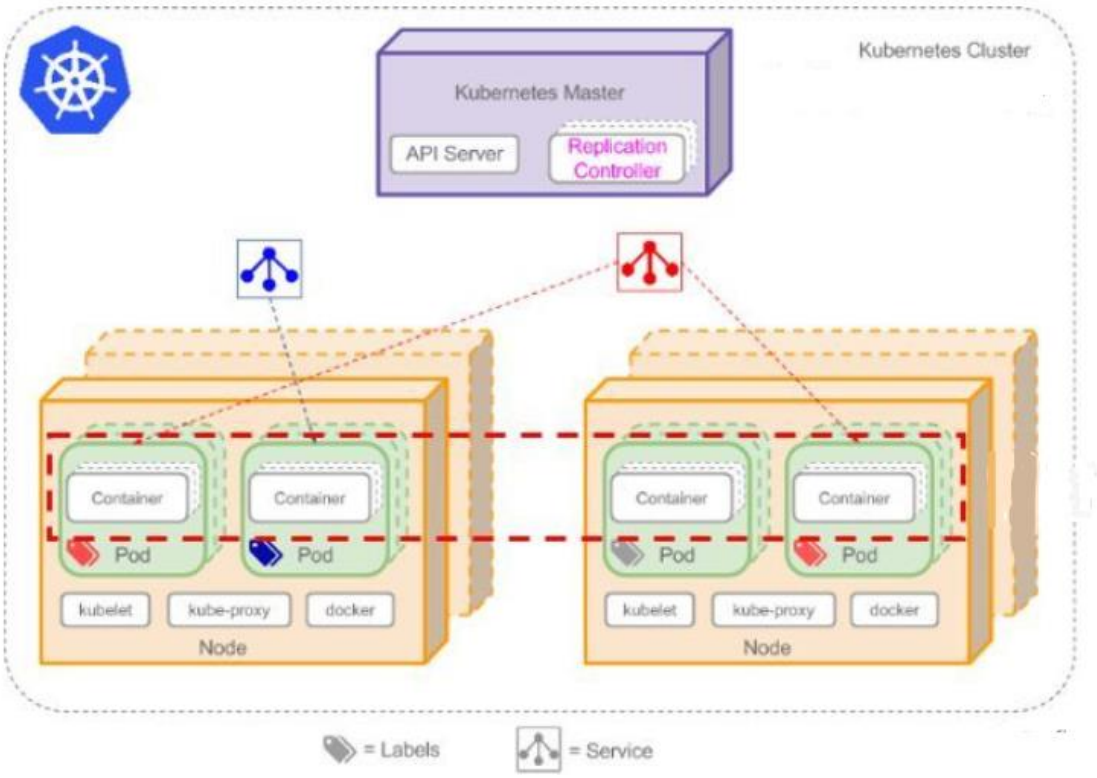

Figure 2 Kubernetes System structure diagram

4. References

[1] Brian Hayes. Cloud Computing[J]. Communications of the Acm, 2008, 51(7):9-11.

[2] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal,等. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility[J]. Future Generation Computer Systems, 25(6):599-616.

[3] Bui, Thanh. Analysis of Docker Security[J]. Computer Science, 2015.

[4] Cito J, Ferme V, Gall H C. Using Docker Containers to Improve Reproducibility in Software and Web Engineering Research[M]// Web Engineering. 2016.

[5] Hüttermann M. Infrastructure as Code[J]. 2012.

[6] Pablo Acuña. Kubernetes[J]. 2016.