CSE 427 Suggestion for End Term Lab Evaluations

Management With vCenter Server

Q1. How to Create a Virtual Machine in HOL ?

Q2. How to create Cloning VMs and using Templates in the Management VCenter server?

Q3. How to Tagg and search to find the objects quickly?

Q4. How to Monitor the events?

Q5. How to create alarms?

Q6. Working with Cloudsim core package?

Q7. CloudSim of all the 8 examples which are done in the Lab classes.

Q9. Run all the commands in Dockers and try to execute all?

Q10. Perform the steps to migrate virtual machines in virtualization 101 lab.

Q12. Create alarm for the CPU utilization in virtualization 101 lab.

References

#vForumOnline: Virtualization 101 Labs Session - YouTube

What's new in vSphere 7.0 - VMware Hands On Labs (HOL) - YouTube

VMware HOL(Hands On Labs) Creating Template and new Virtual Machine from scratch (For Beginners) - YouTube

How to clone a VM to a VM Template on vCenter Server (VCSA) (VMware vSphere ESXi 7) Jason Meers - Bing video

Create Composite Alarms in Amazon CloudWatch - Bing video

Amazon CloudWatch Internet Monitor | Amazon Web Services - Bing video

Amazon CloudWatch Tutorial | AWS Certification | Cloud Monitoring Tools | AWS Tutorial | Edureka - Bing video

Google Cloud Data Catalog - Tags & Tag Templates (Medium article) - YouTube

VMware vSphere: VM Management - Cloning/Templating - Bing video

VIVA Questions

Dockers VIVA Suggestions which you all can follow [Note:- there is no guarantee that this will be coming 100%.. this is a concept suggestion…which you all can go through it]

1. What is Hypervisor?

A hypervisor is a software that makes virtualization possible. It is also called Virtual Machine Monitor. It divides the host system and allocates the resources to each divided virtual environment. You can basically have multiple OS on a single host system. There are two types of Hypervisors:

> Type 1: It’ s also called Native Hypervisor or Bare metal Hypervisor. It runs directly on the underlying host system. It has direct access to your host’ s system hardware and hence does not require a base server operating system.
> Type 2: This kind of hypervisor makes use of the underlying host operating system. It’ s also called Hosted Hypervisor.

2. What is virtualization?

Virtualization is the process of creating a software-based, virtual version of something(compute storage, servers, application, etc.). These virtual versions or environments are created from a single physical hardware system. Virtualization lets you split one system into many different sections which act like separate, distinct individual systems. A software called Hypervisor makes this kind of splitting possible. The virtual environment created by the hypervisor is called Virtual Machine.

3. What is containerization?

Let me explain this is with an example. Usually, in the software development process, code developed on one machine might not work perfectly fine on any other machine because of the dependencies. This problem was solved by the containerization concept. So basically, an application that is being developed and deployed is bundled and wrapped together with all its configuration files and dependencies. This bundle is called a container. Now when you wish to run the application on another system, the container is deployed which will give a bug-free environment as all the dependencies and libraries are wrapped together. Most famous containerization environments are Docker and Kubernetes.

4. Difference between virtualization and containerization

Once you’ ve explained containerization and virtualization, the next expected question would be differences. The question could either be differences between virtualization and containerization or differences between virtual machines and containers. Either way, this is how you respond.

Containers provide an isolated environment for running the application. The entire user space is explicitly dedicated to the application. Any changes made inside the container is never reflected on the host or even other containers running on the same host. Containers are an abstraction of the application layer. Each container is a different application.

Whereas in Virtualization, hypervisors provide an entire virtual machine to the guest(including Kernal). Virtual machines are an abstraction of the hardware layer. Each VM is a physical machine.

5. What is Docker?

Since its a Docker interview, there will be an obvious question about what is Docker. Start with a small definition.

Docker is a containerization platform which packages your application and all its dependencies together in the form of containers so as to ensure that your application works seamlessly in any environment, be it development, test or production. Docker containers, wrap a piece of software in a complete filesystem that contains everything needed to run: code, runtime, system tools, system libraries, etc. It wraps basically anything that can be installed on a server. This guarantees that the software will always run the same, regardless of its environment.

6. What is a Docker Container?

Docker containers include the application and all of its dependencies. It shares the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud. Docker containers are basically runtime instances of Docker images.

7. What are Docker Images?

When you mention Docker images, your very next question will be " what are Docker images" .

Docker image is the source of Docker container. In other words, Docker images are used to create containers. When a user runs a Docker image, an instance of a container is created. These docker images can be deployed to any Docker environment.

8. What is Docker Hub?

Docker images create docker containers. There has to be a registry where these docker images live. This registry is Docker Hub. Users can pick up images from Docker Hub and use them to create customized images and containers. Currently, the Docker Hub is the world' s largest public repository of image containers.

9. Explain Docker Architecture?

Docker Architecture consists of a Docker Engine which is a client-server application with three major components:

1. A server which is a type of long-running program called a daemon process (the docker command).
2. A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
3. A command line interface (CLI) client (the docker command).
4. The CLI uses the Docker REST API to control or interact with the Docker daemon through scripting or direct CLI commands. Many other Docker applications use the underlying API and CLI.

Refer to this blog, to read more about **Docker Architecture**.

10. What is a Dockerfile?

Let’ s start by giving a small explanation of Dockerfile and proceed by giving examples and commands to support your arguments.

Docker can build images automatically by reading the instructions from a file called Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build, users can create an automated build that executes several command-line instructions in succession.

The interviewer does not just expect definitions, hence explain how to use a Dockerfile which comes with experience. Have a look at this tutorial to understand how Dockerfile works.

11. Tell us something about Docker Compose.

Docker Compose is a YAML file which contains details about the services, networks, and volumes for setting up the Docker application. So, you can use Docker Compose to create separate containers, host them and get them to communicate with each other. Each container will expose a port for communicating with other containers.

12. What is Docker Swarm?

You are expected to have worked with Docker Swarm as it’ s an important concept of Docker.

Docker Swarm is native clustering for Docker. It turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts.

13. What is a Docker Namespace?

A namespace is one of the Linux features and an important concept of containers. Namespace adds a layer of isolation in containers. Docker provides various namespaces in order to stay portable and not affect the underlying host system. Few namespace types supported by Docker – PID, Mount, IPC, User, Network

14. What is the lifecycle of a Docker Container?

This is one of the most popular questions asked in Docker interviews. Docker containers have the following lifecycle:

Create a container
Run the container
Pause the container(optional)
Un-pause the container(optional)
Start the container
Stop the container
Restart the container
Kill the container
Destroy the container

15. What is Docker Machine?

Docker machine is a tool that lets you install Docker Engine on virtual hosts. These hosts can now be managed using the docker-machine commands. Docker machine also lets you provision Docker Swarm Clusters.

Docker Basic Commands

Once you’ve aced the basic conceptual questions, the interviewer will increase the difficulty level. So let’s move on to the next section of this Docker Interview Questions article. This section talks about the commands that are very common amongst docker users.

16. How to check for Docker Client and Docker Server version?

The following command gives you information about Docker Client and Server versions:

```
$ docker version
```

17. How do you get the number of containers running, paused and stopped?

You can use the following command to get detailed information about the docker installed on your system.

```
$ docker info
```

Docker Certification Training Course

Explore Curriculum

You can get the number of containers running, paused, stopped, the number of images and a lot more.

18. If you vaguely remember the command and you’d like to confirm it, how will you get help on that particular command?

The following command is very useful as it gives you help on how to use a command, the syntax, etc.

```
$ docker --help
```

The above command lists all Docker commands. If you need help with one specific command, you can use the following syntax:

```
$ docker <command> --help
```

19. How to login into docker repository?

You can use the following command to login into hub.docker.com:

```
$ docker login
```

You’ll be prompted for your username and password, insert those and congratulations, you’re logged in.

20. If you wish to use a base image and make modifications or personalize it, how do you do that?

You pull an image from docker hub onto your local system

It’ s one simple command to pull an image from docker hub:

```
$ docker pull <image_name>
```

21. How do you create a docker container from an image?

Pull an image from docker repository with the above command and run it to create a container. Use the following command:

```
$ docker run -it -d <image_name>
```

Most probably the next question would be, what does the ‘ -d’ flag mean in the command?

-d means the container needs to start in the detached mode. Explain a little about the detach mode. Have a look at this blog to get a better understanding of different docker commands.

22. How do you list all the running containers?

The following command lists down all the running containers:

```
$ docker ps
```

23. Suppose you have 3 containers running and out of these, you wish to access one of them. How do you access a running container?

The following command lets us access a running container:

```
$ docker exec -it <container id> bash
```

The exec command lets you get inside a container and work with it.

24. How to start, stop and kill a container?

The following command is used to start a docker container:

```
$ docker start <container_id>
```

and the following for stopping a running container:

```
$ docker stop <container_id>
```

kill a container with the following command:

```
$ docker kill <container_id>
```

25. Can you use a container, edit it, and update it? Also, how do you make it a new and store it on the local system?

Of course, you can use a container, edit it and update it. This sounds complicated but its actually just one command.

```
$ docker commit <conatainer id> <username/imagename>
```

26. Once you’ ve worked with an image, how do you push it to docker hub?

```
$ docker push <username/image name>
```

27. How to delete a stopped container?

Use the following command to delete a stopped container: