Kubernetes Interview Questions & Answers

By Dharti Sutariya

Author and DevOps Engineer

DotNetTricks

Kubernetes Interview Questions & Answers

The author of this book has tried their best to ensure the accuracy of the information described in this book. However, the author cannot guarantee the accuracy of the information contained in this book. The author or Dot Net Tricks Innovation Pvt. Ltd. will not be liable for any damages, incidental or consequential caused directly or indirectly by this book.

Further, readers should be aware that the websites or reference links listed in this book may have changed or disappeared between when this book was written and when it is read.

Release History

- Initial Release 1.0 - 3rd Nov 2020

About Dot Net Tricks

Dot Net Tricks was founded by Shailendra Chauhan (Microsoft MVP), in Jan 2010. Dot Net Tricks came into existence in the form of a blog post over various technologies including .NET, C#, SQL Server, ASP.NET, ASP.NET MVC, JavaScript, Angular, Node.js and Visual Studio, etc.

The company which is currently registered by a name of Dot Net Tricks Innovation Pvt. Ltd. came into the shape in 2015. Dot Net Tricks website has an average footfall on the tune of 300k+ per month. The site has become a cornerstone when it comes to getting skilled-up on .NET technologies and we want to gain the same level of trust in other technologies. This is what we are striving for.

We have a very large number of trainees who have received training from our platforms and immediately got placement in some of the reputed firms testifying our claims of providing quality training. The website offers you a variety of free study material in the form of articles.

Dot Net Tricks Courses

Master in-demand job skills with our step by step and project-based courses. Learn to start a new career, with our curated learning paths tailored to today's developers and technology needs. Learn to code, prepare yourself for interviews, and get hired!

We offer the courses in the following categories:

- .NET Development
- Front-end Development
- Cloud
- DevOps
- Programming Languages
- Database - SQL and NoSQL
- Mobile Development
- ML/AI and many more...

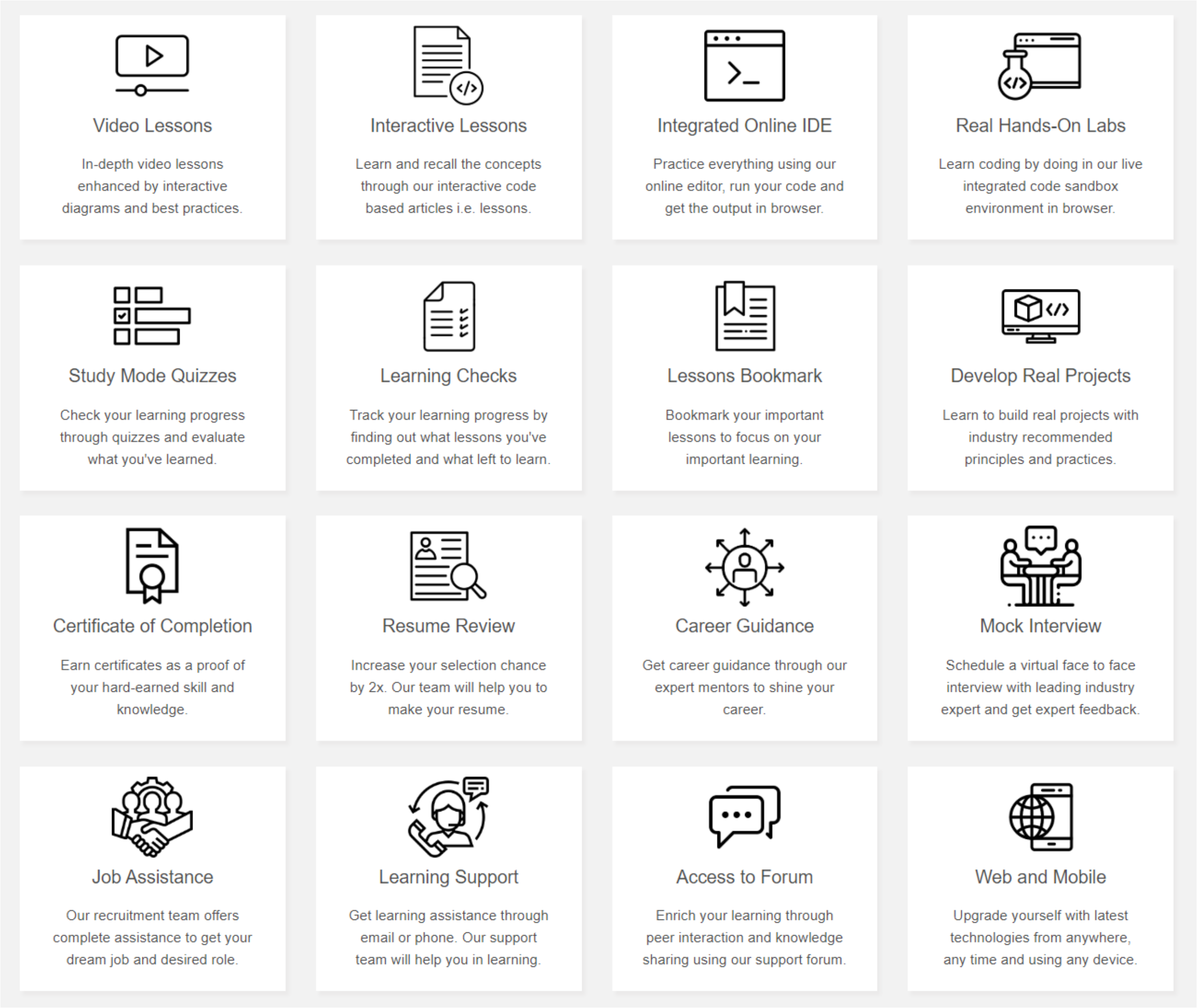
You can start learning free from here: https://www.dotnettricks.com/courses

Dot Net Tricks Plus

DotNetTricks Plus unlocks the access of DotNetTricks premium features like unlimited access to all courses, source codes, and assessments. Get help over email or phone. Upgrade your skills with curated learning paths tailored to today's developers and technology needs. Learn new skills and discover the world of possibilities with step-by-step guidance.

Dot Net Tricks Plus Features

Each Plus member will get access to our full-featured learning platform with advanced tools, which is required to learn new skills and advance his/her career.



Video Lessons

In-depth video lessons enhanced by interactive diagrams and best practices.

Interactive Lessons

Learn and recall the concepts through our interactive code based articles i.e. lessons.

Integrated Online IDE

Practice everything using our online editor, run your code and get the output in browser.

Real Hands-On Labs

Learn coding by doing in our live integrated code sandbox environment in browser.

Study Mode Quizzes

Check your learning progress through quizzes and evaluate what you've learned.

Learning Checks

Track your learning progress by finding out what lessons you've completed and what left to learn.

Lessons Bookmark

Bookmark your important lessons to focus on your important learning.

Develop Real Projects

Learn to build real projects with industry recommended principles and practices.

Certificate of Completion

Earn certificates as a proof of your hard-earned skill and knowledge.

Resume Review

Increase your selection chance by 2x. Our team will help you to make your resume.

Career Guidance

Get career guidance through our expert mentors to shine your career.

Mock Interview

Schedule a virtual face to face interview with leading industry expert and get expert feedback.

Job Assistance

Our recruitment team offers complete assistance to get your dream job and desired role.

Learning Support

Get learning assistance through email or phone. Our support team will help you in learning.

Access to Forum

Enrich your learning through peer interaction and knowledge sharing using our support forum.

Web and Mobile

Upgrade yourself with latest technologies from anywhere, any time and using any device.

Start your journey today to learn coding. Because learning to code is the first step and forward to advance your career.

The detail about DotNetTricks Plus can be found here: https://www.dotnettricks.com/plus-membership

Corporate Training

Dot Net Tricks having a pool of mentors who help the corporate to enhance their employment skills as per changing the technology landscape. Dot Net Tricks offers customized training programs for new hires and experienced employees through online and classroom mode. As a trusted and resourceful training partner, Dot Net Tricks helps the corporate to achieve success with its industry-leading instructional design and customer training initiatives.

Apart from these, we also provide on-demand boot camps and personalized project consultation.

The detail about Corporate Training can be found here: https://www.dotnettricks.com/corporate-training

Dot Net Tricks Unlimited Live Training

Instructor-led Live Online Training Programs

DotNetTricks unlimited membership unlock unlimited access to all courses, learning paths, study mode quizzes, hands-on labs, source code and study material. Upgrade your skills with our structured learning paths tailored to today's developers and technology needs. DotNetTricks Unlimited Live training membership includes the **benefits of DotNetTricks Plus** membership and **Unlimited live online training**.

DotNetTricks Unlimited Live Training enables you to Become:

- **Full-stack JavaScript Developer** - Angular, ReactJS, Node.js
- **Full-stack .NET Developer** - .NET, ASP.NET Core, MVC5, WebAPI
- **Technical Architect** - Microservices, Design Patterns and Clean Architecture
- **Cloud Engineer/Architect** - Microsoft Azure, AWS
- **DevOps Engineer** - DevOps, Docker and Kubernetes
- **Mobile Developer** - Xamarin Forms, React Native

The detail about Unlimited Live Training can be found here: https://www.dotnettricks.com/instructor-led-courses

Dot Net Tricks eBooks

Dot Net Tricks offer a wide range of eBooks on technical interviews Q&A. All eBooks are written by industry experts and coaches. These eBooks will help you to prepare yourself for your next job within a short time. We offer the eBooks in the following categories:

- .NET Development
- Front-end Development
- Cloud
- DevOps
- Programming Languages
- Database - SQL and NoSQL
- Mobile Development
- ML/AI and many more...

You can buy other eBooks from here: https://www.dotnettricks.com/books

Join us today, learn to code, prepare yourself for interviews, and get hired!

Dedication

I would like to say many thanks to my father who gave me the opportunity to make my carries as I wanted when I was in 10th standard. I gave the speech about “Beti bachao” and I got national awards for the first time. I have lots of expertise on DevOps and one day Ankur Mistry saw my technical knowledge and he inspired me to write a book and share the knowledge to others. The Next day I started the book and also, I believe reading and writing is the most important part of gaining knowledge in life.

I would like to say thanks to my father and mother for teaching me nothing is impossible and be the best version of yourself.

Also, thanks to my heroes(brothers) Vipul and Pankaj, who always remind me who I am and what I can do, always support me no matter what situation.

Special thanks to Ankur Mistry who is my inspiration.

To my wonderful readers, may you find this book helpful?

“Keep your mind open to change all the time. Welcome it. Always go with the choice that scares you the most, because that’s the one that is going to help you grow”

With lots of love.

-Dharti Sutariya

Introduction

What Where Author Qualification to Write This Book

Dharti Sutariya is DevOps Engineer. She has more than 3 years of experience on DevOps technologies such as Explicit Technical Knowledge and hands-on experience of DevOps processes and practices, Automation, Auto scaling, Monitoring, and Configuration Management. Gained expertise in SDLC and Agile implementation. Experienced in maintaining the Continuous flow from Code phase to Deployment and Production Phase. Experienced DevOps Model, CI/CD with Jenkins and Azure DevOps 2019 server.

What This Book Is

Kubernetes, a powerful tool in DevOps field, was originally designed by Google and is now maintained by the Cloud Native Computing Foundation. Automates the application deployments, scaling the cluster and container, scheduling, maintenance, and operation of multiple application containers across clusters of nodes.

What You'll Learn

This book is for the techy, Freshers and Experienced DevOps Engineers who are looking for a change or want to make a bright future in DevOps. This book covers the interview questions on the following topics:

- Container Orchestration
- Fundamentals of Kubernetes
- Minikube
- Kubernetes Architecture
- Kubernetes Object and Workload
- Kubernetes Master Components
- Kubernetes Node Components
- Kubernetes Installation
- Kubernetes Commands
- Kubernetes Service
- Kubernetes Pods
- Kubernetes Deployments
- Kubernetes Networking

Our best wishes are always with you for your learning and growth!

About the Author

Dharti Sutariya - DevOps Engineer and Author

She achieved great technical knowledge in a short period of time in DevOps Technology, she has vast experience in DevOps area. She graduated from Gujarat Technology University (GTU)- India. She has an ability to set up a fully DevOps oriented process.

She is working as a DevOps Engineer in an MNC with her major interest in technologies and frameworks like Docker, Kubernetes, Terraform, Ansible, Chef and many more using platform Linux and Windows.

She has explicit Technical Knowledge and hands-on experience of DevOps processes and practices, Automation, Auto scaling, Monitoring, and Configuration Management. Gained expertise in SDLC and Agile implementation. Experienced in maintaining the Continuous flow from Code phase to Deployment and Production Phase. Experienced DevOps Model, CI/CD with Jenkins and Azure DevOps 2019 server.

How to Contact Us

Although the author of this book has tried to make this book as accurate as it possible but if there is something strikes you as odd, or you find an error in the book please drop a line via e-mail.

The e-mail addresses are listed as follows:

- mentor@dotnettricks.com
- info@dotnettricks.com

We are always happy to hear from our readers. Please provide your valuable feedback and comments!

You can follow us on YouTube, Facebook, Twitter, LinkedIn and Google Plus or subscribe to RSS feed.

Table of Contents

1

Container Orchestration

Q1. What is a Container?

Ans. A container is an executable unit. It is small, as it does not contain device drivers like other machines. Containers do not contain OS images due to this it is platform-independent, portable and lightweight.

Q2. Why do we need containers?

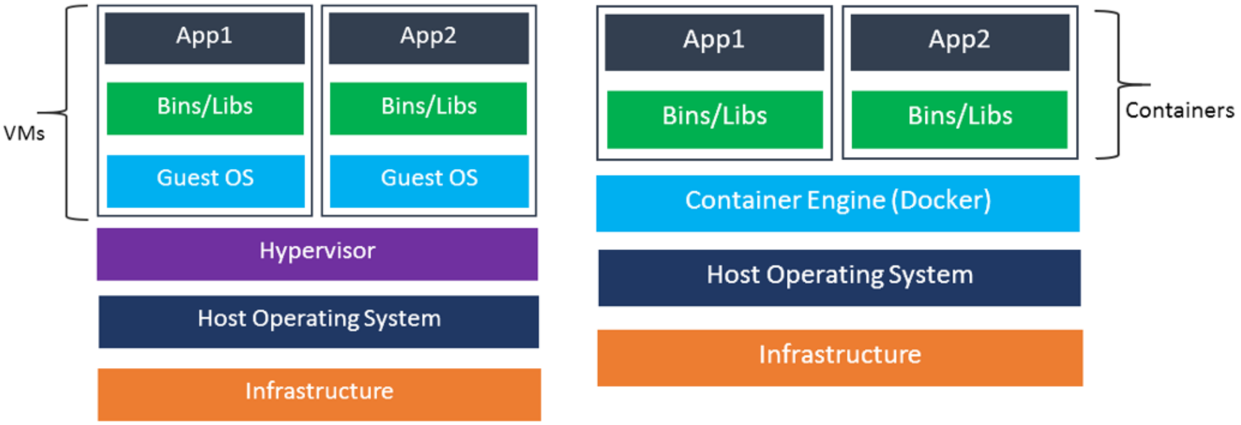
Ans. We Need container because:

- A container gives the ability for predictable environments which is isolated from other applications and environments.
- Also contain necessary dependency of the application like version, software libraries.
- we can break down complex applications into smaller ones.
- We can scale application as per need

Q3. What is the difference between containers and virtual machines?

Ans. The differences are given below:

Virtual Machine	**Container**
Each VM run its OS	All container shares the same kernel of the host
Required less memory space	Required more memory space
Provide process level isolation, which is less secure	Fully isolated and are more secure
Boot up time is in minutes	Container initiate in a second
Heavyweight	Lightweight
No Version Controlled	Images can be diffed and can be version controlled
Need hypervisor	Don’t need a hypervisor



Q4. Which types of platforms are supported by containers?

Ans. Containers support the following platforms:

1. Windows 10
2. Windows Server 2016(Native container support)
3. Windows Server 2019(Native container support)
4. Mac OS (Native container support)
5. Linux
6. Azure
7. Amazon AWS
8. Google Cloud

Q5. What are the benefits of containers?

Ans. The benefits of containers are:

1. Managing network interfaces and applying resources inside containers.
2. There is less code needed to transfer and upload the workloads
3. Consistent Environment: “Isolating applications and operating systems through containers”.
4. Platform independence: Build it once, run it anywhere: able to run virtually anywhere, on Linux, Windows, and Mac operating systems; on virtual machines or bare metal”
5. Effectual isolation and resource sharing
6. We can control container functionality like Speed: Start, create, replicate or destroy containers in seconds
7. Immense and smooth scaling
8. Improved developer productivity and development pipeline

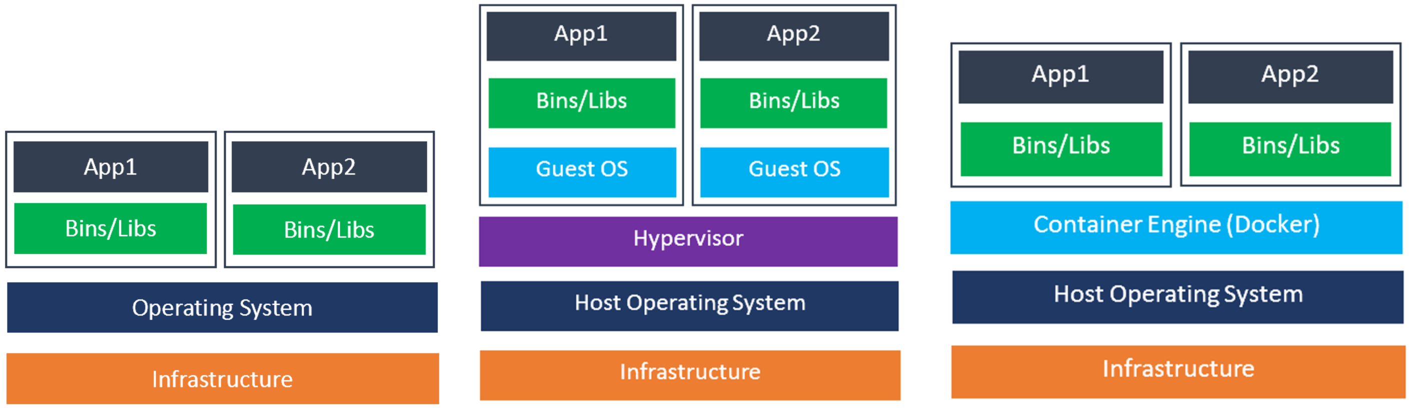
Q6. Are there any disadvantages or Limitation of containers?

Ans. Yes, there is some limitation of containers which we need to keep in mind:

1. All containers are running via the host system’s kernel
2. Only allows Linux “guest” operating systems.
3. A container is not a full virtualization stack like Xen, KVM or libvirt.
4. Mostly Security depends on the host system; hence, containers are not secure.

Q7. What is the difference between deploying applications on hosts and containers?

Ans. When running an application as a container only you need an image with dependencies of your application and while deploying directly on the host before hosting the app you need to install dependencies on host this might be effective on existing installed dependencies.



Bare Metal
(Deploy in Days/Weeks)

Virtual Machines
(Deploy in Hours/Days)

Containers
(Deploy in Seconds/Minutes)

Q8. What is Container Orchestration?

Ans. Container orchestration is one automatic process which manages the container life cycle.

Q9. What is container orchestration used for?

Ans. container orchestration used for:

1. Redundancy and availability of containers
2. Scaling up or removing containers to spread application load in a balanced or impartial way across host infrastructure
3. Movement of containers from one host to another if there is a shortage of resources in a node, or if a node dies.
4. Allotment of resources between containers
5. External exposure of services running in a container with the outside world
6. Load balancing between containers with API gateway
7. Health monitoring of containers and hosts
8. Configuration of an application about the containers running it

Q10. Mention the various container resource monitoring tools?

Ans. The various container monitoring tools are as follows:

a. Grafana
b. Heapster
c. CAdvisor
d. InfluxDB
e. Prometheus

Q11. What are various Container orchestration tools?

Ans. There are so many tools for Container orchestration, here I have listed down those tools.

1. Kubernetes
2. Docker Swarm,
3. Nomad
4. Docker Compose
5. Minikube
6. Cloudify-Pure-play Cloud Orchestration & Automation
7. OpenShift
8. Rancher
9. Containership
10. Marathon on Apache Mesos
11. Amazon ECS
12. AZK
13. GiantSwarm
14. Lunchbadger
15. Google Container Engine

2

Fundamentals of Kubernetes

Q1. What is Kubernetes?

Ans. Kubernetes is Google's open-source system(tool) for managing Linux and Windows containers across private, public and hybrid cloud environments. In short, it is a container management tool.

Kubernetes automates the application deployments, scaling the cluster and container, scheduling, maintenance, and operation of multiple application containers across clusters of nodes.

Kubernetes contains tools for orchestration, service discovery and load balancing that can be used with different containerized systems like Docker and Rocket containers.

Q2. Why use Kubernetes?

Ans. As per growing IT industries in the market microservice has been introduced and it is more powerful than monolithic services.

Deploying applications in containers for production environments, if one container goes down another container needs to start, but keep in mind you need to manage the containers that run the applications with zero downtime. Wouldn’t it be easier if this behaviour was managed by a Kubernetes? That is why we need Kubernetes for hundreds of micro and monolithic applications.

Q3. What are the benefits of Kubernetes?

Ans. There are a lot of benefits to using Kubernetes as an autoscaling cluster.

- Control and automate deployments and updates
- It’s portable and 100% open source
- Monitoring and self-healing abilities
- Intelligent and balanced scheduling of containers
- Monitoring of node and cluster health
- Test and autocorrection of applications
- Automated rollouts and rollbacks
- Horizontal infrastructure scaling
- Health checks and self-healing
- Traffic routing and load balancing
- DNS management
- Storage orchestration (automatically mount the necessary storage system, be it local storage, or public cloud provider storage)

- Zero Downtime with Rolling Deployments
- Secrets Management
- CI/CD Pipelines
- Provides Cronjob objects that can complete a task at a certain time
- Effective migration to the cloud
- Multi-cloud (and hybrid cloud) flexibility

Q4. What are different types of objects available in Kubernetes?

Ans. Following main objects are available in Kubernetes:

- Pod
- Node
- Service
- Replica Set
- Namespace
- Replication Controller
- Deployments
- Volume
- Secret
- kubectl

Q5. What are Kubernetes alternatives?

Ans. The Kubernetes alternatives are:

- Rancher
- Nomad
- Docker Swarm
- Cloud Foundry
- AWS Fargate
- Apache Mesos
- OpenStack
- Docker Compose.

Q6. What is the difference between Kubernetes and docker swarm?

Ans. The difference between Kubernetes and Docker Swarm is given below:

Feature	Kubernetes	Docker-Swarm
Installation	Easy and fast to install and configure	Takes some work to get up and running
GUI support	Provide Web UI (Dashboard)	Need to create Image for dashboard or we can use “docker run -it --rm -p 8080:8080 -v /var/run/docker.sock:/var/run/docker.sock julienbreux/docker-swarm-gui:latest”
Cluster configuration	In Kubernetes, installation is manual and it takes serious planning to make Kubernetes up and running. Installation is different from OS to OS	Docker Swarm is simple to install as compared to Kubernetes, The other

	provider to provider. In kubernetes, which node takes what role is needed to know before applying a resource to the cluster.	functionality is we can add worker and manager nodes after configuration.
API Functionality	Kubernetes uses its own client, API and YAML definitions which each differ from that of the standard Docker equivalents	The Swarm API provides much of the familiar functionality from Docker itself but does not enclose all of its commands
Scalability	Quick container deployment and scaling even in very large clusters	Provides strong guarantees to cluster states at the expense of speed
High Availability	In Kubernetes, you can configure Multiple master nodes and for High Availability, you can replicate important components on master nodes for HA Environments.	Docker Swarm also provides high availability as services can be replicated in Swarm nodes. In Docker Swarm, Swarm manager nodes are responsible for the whole cluster and manage the resources.
Load Balancing	Automated internal load balancing through any node in the cluster	Enabling load balancing requires manual service configuration
Container updates and rollbacks	Kubernetes handles the update process progressively monitoring service health to retain availability throughout the update process making changes to one pod at the time preventing a service outage.	Docker Swarm container updates are done by informing the scheduler to use a new image instead. The update can be rolled out if something goes wrong.
Logging and monitoring	Kubernetes supports multiple levels of logging, monitoring and debugging when the resources are deployed within the cluster	Docker Swarm is supported for only monitoring with the third-party tools. Docker Swarm has an open API, it makes it easier to connect with plenty of apps.
Networking	Kubernetes uses flannel, Calico, Weave Net and others to accomplish container networking. Containers are joined in this network and announced through etcd.	Docker Swarm forms a multi-host ingress network overlay that connects containers running on all cluster nodes. Inter-container networks can be configured manually. Connections between nodes are also automatically secured through TLS authentication with certificates.
Data volumes	Volumes shared within pods	Simple shared local volumes
Service Discovery	Containers can be defined as easily discoverable services	Services are discoverable throughout the cluster network

Q7. Can we use Kubernetes as an alternative to docker?

Ans. No, we can't use Kubernetes as an alternative to docker, because docker is used for containerization and Kubernetes is used for container autoscaling.

Q8. Can we install Kubernetes on the Windows platform?

Ans. **Yes,** we can install minikube in windows platform locally but it is for single-node clusters**.** We can't use minikube in our production environment.

If we want to install Kubernetes for production environment then we can't install it in windows platform as Kubernetes master node, it is only supported in Linux platform.

Q9. Can we autoscale windows containers using Kubernetes?

Ans. Yes, we can, but first, you need to configure the windows node where your container is running to Kubernetes cluster which is on Linux platform.

Q10. Which platform is supported by kubernetes-master?

Ans. Only Linux platforms are supported. In Linux we can use below:

- Ubuntu 16.04+
- Debian 9+
- CentOS 7
- Red Hat Enterprise Linux (RHEL) 7
- Fedora 25+
- HypriotOS v1.0.1+
- Container Linux

Q11. Which windows worker node is supported by Kubernetes?

Ans. As per this time 12 April 2020 recently they announced the windows container support in Kubernetes cluster (Linux). If we want to configure our windows worker node to Kubernetes master then we have to keep in mind below points:

K8 Version	Windows OS Version (Node)	Docker Version	Supported
V1.14	Windows Server 1709	Docker EE	Not supported
V1.14	Windows Server 1803	Docker EE	Not supported
V1.14	Windows Server 1809/Windows Server 2019	EE-basic 18.09	Supported for Windows Server containers Builds 17763.* with Docker
V1.18 alpha	Windows CSI CSI drivers (e.g. Azure Disk, GCE PD, etc.)	Docker EE	supported containerized CSI node plugins in Windows via new CSIProxy

Q12. Is there any limitation of Kubernetes?

Ans. Yes, there is some limitation of Kubernetes like:

- Kubernetes control plane only support in Linux platform
- only Kubernetes V1.14 is support windows server 2019
- supports clusters with up to 5000 nodes
- Supports 150000 total pods
- supports 300000 total containers
- supports 100 pods per node

- 2 GiB or more of RAM per machine for Kubernetes
- At least 2 CPUs on the machine that you use as a control-plane node
- Kubernetes allows administrators to set quotas, in namespaces

Q13. Is Kubernetes secure?

Ans. We can say yes because Kubernetes uses TLS (Transport Layer Security) for all API traffic in the cluster, all API communication in the cluster is encrypted by default with TLS. When we install Kubernetes it also installs a majority of certificates to protect the cluster.

Q14. How can we protect Kubernetes?

Ans. Kubernetes is not designed for security purpose even though they provide basic RBACs and infrastructure security features, if we want to make them more secure then we can integrate Kubernetes security tools like "**Run-Time Kubernetes Security – The NeuVector Multi-Vector Container Firewall**".

Q15. Does Kubernetes do load balancing?

Ans. Yes, Kubernetes do load balancing by using two methods through "kube-proxy". The default mode of kube-proxy is "iptables mode" and other most Genuine Load Balancing is "ingress"

Q16. What are the disadvantages of Kubernetes?

Ans. There is some drawback of Kubernetes:

- It is very complex
- It is hard to estimate how much effort is required to implement on a large scale of the software.
- It is more expensive than its alternatives
- Its network configuration is very hard with Windows OS
- It requires experience and extensive training for its debugging and troubleshooting in due time

Q17. Which programming language is Kubernetes written in?

Ans. Kubernetes is written in Go/Golang.

Q18. Can Kubernetes work without Docker?

Ans. Yes, it is vice versa, Kubernetes can run without Docker and Docker can function without Kubernetes. But Kubernetes can (and does) benefit greatly from Docker and vice versa.

3

Minikube

Q1. What is Minikube?

Ans. Minikube is a small VM with a small docker registry by default, which provides a single-node Kubernetes cluster inside the virtual machine. Also, we can call it a “local Kubernetes engine”

Q2. Do you explain the features of minikube?

Ans. Kubernetes supports a lot of features but among them, minikube support below feature:

- DNS
- NodePorts
- ConfigMaps and Secrets
- Dashboards
- Container Runtime: Docker, CRI-O, and containerd
- CNI (Container Network Interface)
- Ingress

Q3. What is the use of Minikube?

Ans. Minikube is used for single-node clusters in a local environment for developer purposes. It is working on Linux, Mac OS and Windows platforms.so we can practice Kubernetes cluster on local machines.

In short, we use minikube as a:

- Local Kubernetes
- Load Balancer
- Multi-cluster

Q4. What are the disadvantages of Minikube?

Ans. The disadvantages are described below:

- Only for a single-node cluster.
- Installation and removal are not as efficient as other solutions.
- Can conflict with a local installation of other tools (such as VirtualBox for win and mac).
- The limited features of Kubernetes are supported.

Q5. What are the advantages of Minikube?

Ans. Advantages are described below:

- The best solution for developer and beginners
- Works on Windows (any version and edition), Mac and Linux
- Very easy and powerful
- Multiple drivers that can match any environment
- Can work with or without an intermediate VM on Linux (vmdriver=none)
- Installs several plugins (such as dashboard) by default
- Very flexible on installation requirements, upgrades and deletion

Q6. What is the difference between Kubernetes and minikube?

Ans. The differences are given below:

Features	Kubernetes	minikube
OS support	Control plane is on Linux platform	Windows, Mac, Linux
Container	Linux, Windows- on the specific version	both
Installation/Configuration	Very complex	Easy to install. For windows - Hyper-V or VirtualBox is required
Network Configuration	Complex	easy to connect with the host network
Load balancing	Very powerful load balancing across the multi-cluster	Has limited support
Cluster	Support multi-cluster	support single-node cluster

Q7. Can you use Minikube in production?

Ans. No, you can't use minikube for production environments because minikube is designed for a single-node cluster and does not run the containers on the host. Which may lead to performance issues on the productions and it is designed for developer purposes. Instead, it uses a "hypervisor", e.g. VirtualBox, VMware Fusion or KVM depending on the platform.

Q8. Does Minikube require Docker?

Ans. Yes, for running containers on a cluster, you need a containerization tool. The recommended tool is docker.

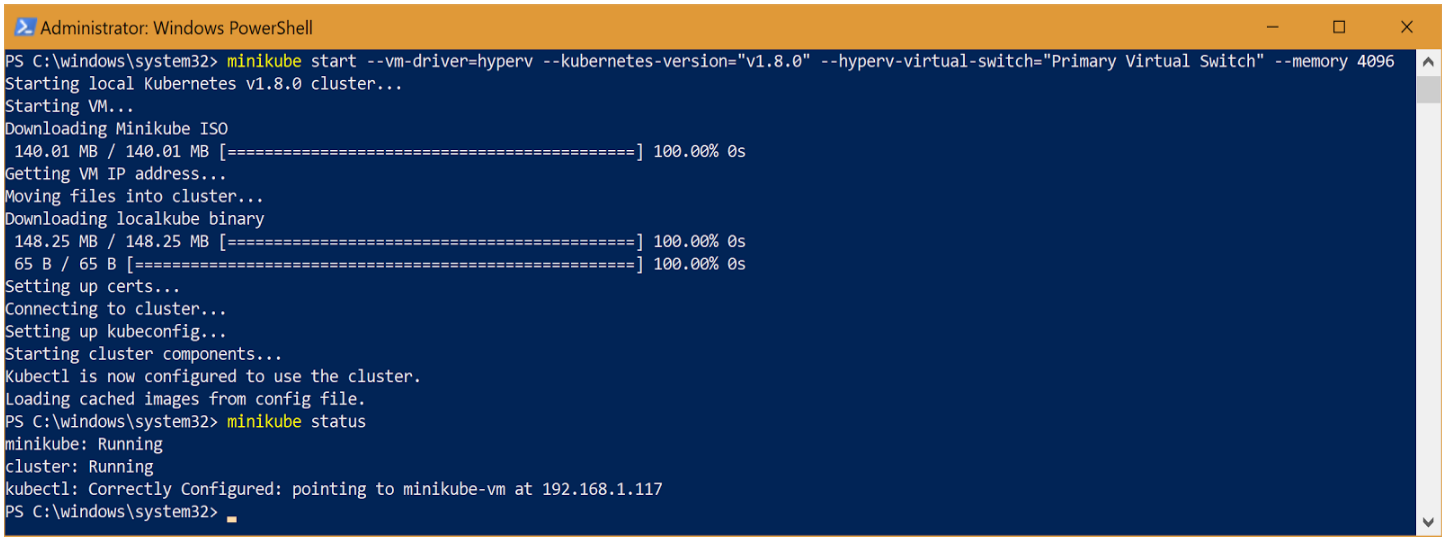
Q9. What command-line interface is used to interact with the Kubernetes cluster?

Ans. Kubectl - The command-line interface for Kubernetes

Q10. How do I Install Minikube on Windows?

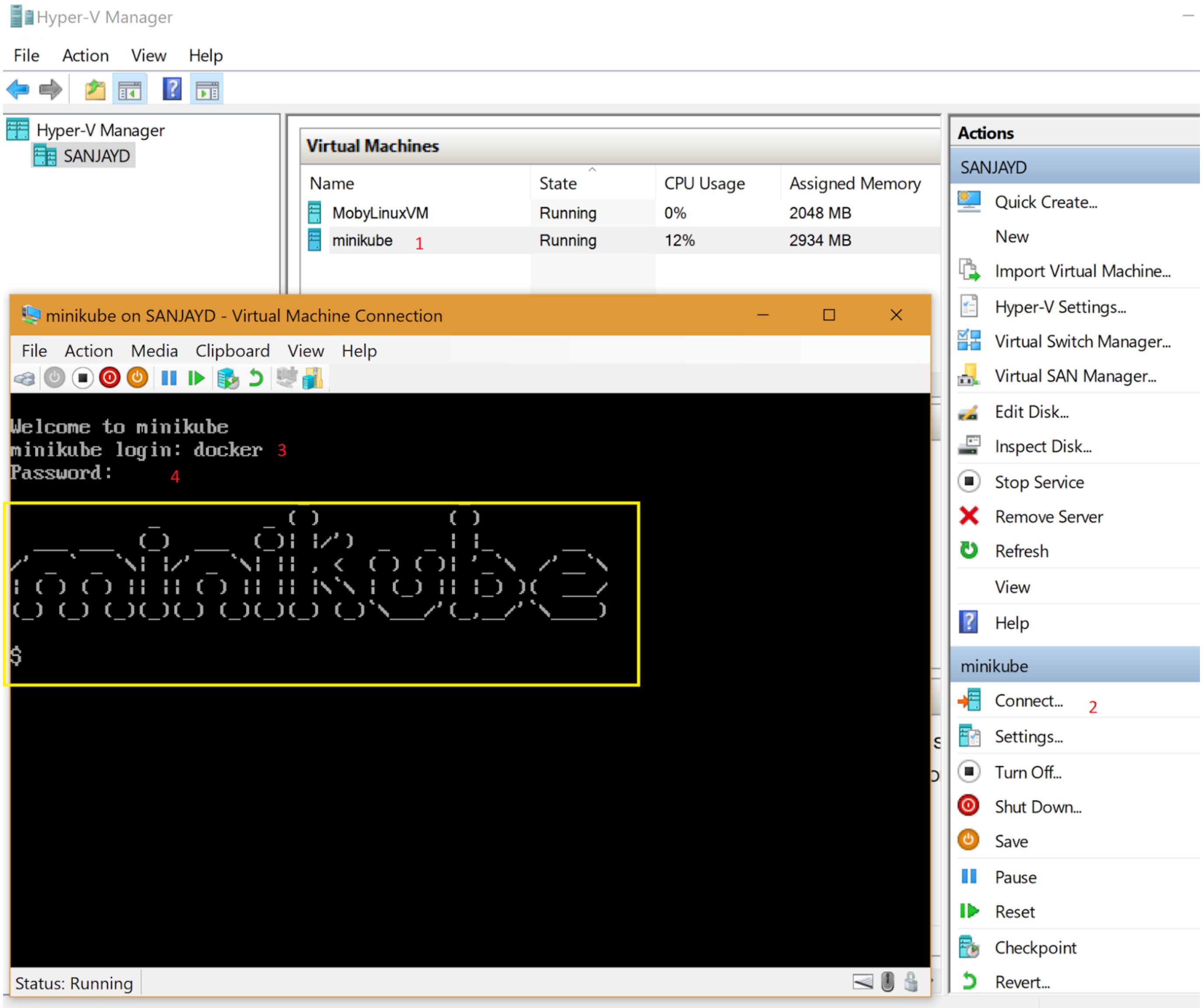
Ans. You can install Minikube on windows with two different ways:

- Make sure you have kubectl installed (order to start working on a Kubernetes cluster, it is necessary to install the Kubernetes command line (**kubectl**))
- Install **VirtualBox** or **Hyper-V** first
- Download the minikube-installer.exe file from git repo
- Add C:\Program Files (x86)\Kubernetes\minikube or similar to the PATH (in System Settings > Environment Variables)
- To confirm successful installation please run the following commands:
  - **minikube start --driver=<driver_name>**
  - Note: **<driver_name>** is enter the name of the hypervisor you installed in lowercase letters
- Once minikube start run the command below to check the status of the cluster
  - minikube status



```
PS C:\windows\system32> minikube start --vm-driver=hyperv --kubernetes-version="v1.8.0" --hyperv-virtual-switch="Primary Virtual Switch" --memory 4096
Starting local Kubernetes v1.8.0 cluster...
Starting VM...
Downloading Minikube ISO
 140.01 MB / 140.01 MB [============================================] 100.00% 0s
Getting VM IP address...
Moving files into cluster...
Downloading localkube binary
 148.25 MB / 148.25 MB [============================================] 100.00% 0s
 65 B / 65 B [======================================================] 100.00% 0s
Setting up certs...
Connecting to cluster...
Setting up kubeconfig...
Starting cluster components...
Kubectl is now configured to use the cluster.
Loading cached images from config file.
PS C:\windows\system32> minikube status
minikube: Running
cluster: Running
kubectl: Correctly Configured: pointing to minikube-vm at 192.168.1.117
PS C:\windows\system32>
```

- The output from **minikube status**
  - host: Running
  - kubelet: Running
  - apiserver: Running
  - kubeconfig: Configured



Name	State	CPU Usage	Assigned Memory
MobyLinuxVM	Running	0%	2048 MB
minikube	Running	12%	2934 MB

```
Welcome to minikube
minikube login: docker
Password:
                         _             _
            _         _ ( )           ( )
  ___ ___  (_)  ___  (_)| |/')  _   _ | |_      __
/' _ ` _ `\| |/' _ `\| || , <  ( ) ( )| '_`\  /'__`\
| ( ) ( ) || || ( ) || || |\`\ | (_) || |_) )(  ___/
(_) (_) (_)(_)(_) (_)(_)(_) (_)`\___/'(_,__/'`\____)

$
```

Reference link: https://kubernetes.io/docs/tasks/tools/install-minikube/

Q11. How do I install Minikube on Mac?

Ans. Installation:

- Check if virtualization is supported
  - sysctl -a | grep -E --color 'machdep.cpu.features|VMX' the output (should be colored)
- Make sure you have kubectl installed
- Install a Hypervisor one of them: HyperKit,VirtualBox,VMware Fusion
- Install Minikube
  - brew install minikube

- You can also install it on macOS by downloading a stand-alone binary:
    - curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-darwin-amd64 \ && chmod +x minikube
- Here’s an easy way to add the Minikube executable to your path:
    - sudo mv minikube /usr/local/bin

Reference link: https://kubernetes.io/docs/tasks/tools/install-minikube/

Q12. How do I install Minikube on Linux?

Ans. The steps are given below:

- Make sure you have installed kubectl
- Install a Hypervisor
    - KVM, which also uses QEMU
    - VirtualBox
- Install Minikube via direct download:
- curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 \ && chmod +x minikube
- sudo mkdir -p /usr/local/bin/
- sudo install minikube /usr/local/bin/
- install Minikube using Homebrew
    - brew install minikube

Reference link: https://kubernetes.io/docs/tasks/tools/install-minikube/

Q13. How do I install a Helm in Minikube?

Ans. Helm is a Kubernetes Package Manager, it is a tool for managing Kubernetes charts. Using helm, you can manage Kubernetes manager applications. Helm provides different types of features that are useful for managing applications, like - inspect, upgrade, install and delete.

you can install helm in your minikube cluster and utilize its features.

For Windows:

- you can download the latest tar file with the Windows binary from Releases · helm/helm and extract its contents.
- Copy helm.exe to the bin directory
- Run helm init from the command line to install "tiller" in the Minikube cluster.

For Linux:

- **Snap (Linux):** sudo snap install helm --classic

- **Script:** Helm now has an installer script that will automatically grab the latest version of Helm and install it locally. You can fetch that script and then execute it locally.

  ```
  $ curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3

  $ chmod 700 get_helm.sh

  $ ./get_helm.sh
  ```

Q14. How do I get rid of Minikube for windows?

Ans. You need to perform some command in your PowerShell

```
minikube stop & REM stops the VM
minikube delete & REM deleted the VM

Then delete the .minikube and .kube directories commonly under:
C:\users\{user}\.minikube
```

Q15. How do I get rid of Minikube for ubuntu?

Ans. You need to perform some commands:

```
kubeadm reset
sudo apt-get purge kubeadm kubectl kubelet kubernetes-cni kube*
sudo apt-get autoremove
sudo rm -rf ~/.kube

And restart the computer.
```

Q16. What is Minikube, Kubectl and kubelet?

Ans. **Minikube** - k8s cluster with a single node. It is a local environment for the developer or administrator to test and develop applications locally.

- **kubectl** - command-line interface for k8s cluster. you can manage the Kubernetes resources like create, delete, update components and debug stuff in k8s using kubectl.
- **kubelet** - k8s process that runs on each node to manage containers: starting, communicating with them etc. Its job is to assure that a pod spec provided by the API server is healthy.

Q17. Does Minikube support Load Balancer?

Ans. yes. minikube supports load balancer but there is some limitation for that it cannot support external load balancers. load balancer service is working fine with minikube.

Q18. How do I start Minikube in Ubuntu?

Ans. follow this command:

```
$ sudo apt-get update
```

$ sudo apt-get install -y apt-transport-https

$ sudo apt-get install -y virtualbox virtualbox-ext-pack

$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

$ sudo touch /etc/apt/sources.list.d/kubernetes.list

$ echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee -a /etc/apt/sources.list.d/kubernetes.list

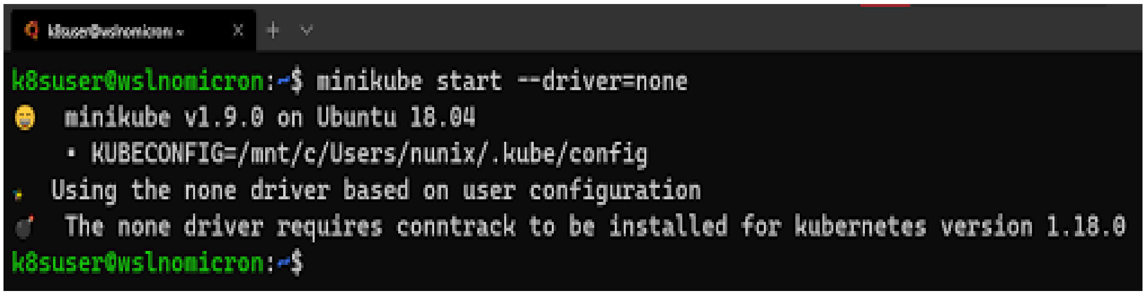
$ sudo apt-get update

$ sudo apt-get install -y kubectl

$curl-Lo minikube https://storage.googleapis.com/minikube/releases/v0.28.2/minikube-linux-amd64

$ chmod +x minikube && sudo mv minikube /usr/local/bin/

$ minikube start



```
k8suser@wslnomicron:~$ minikube start --driver=none
   minikube v1.9.0 on Ubuntu 18.04
    ▪ KUBECONFIG=/mnt/c/Users/nunix/.kube/config
  Using the none driver based on user configuration
   The none driver requires conntrack to be installed for kubernetes version 1.18.0
k8suser@wslnomicron:~$
```

$ kubectl api-versions

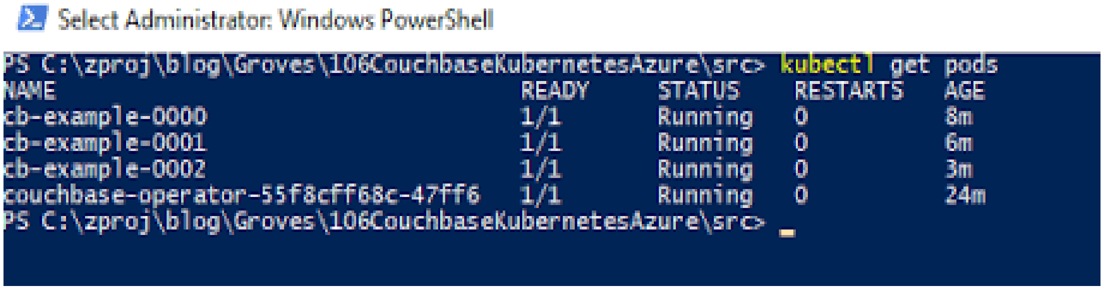
Q19. How do you make a pod on Minikube?

Ans. Create the Pod:

```
kubectl apply -f <your yml file>
```

Verify that is running:

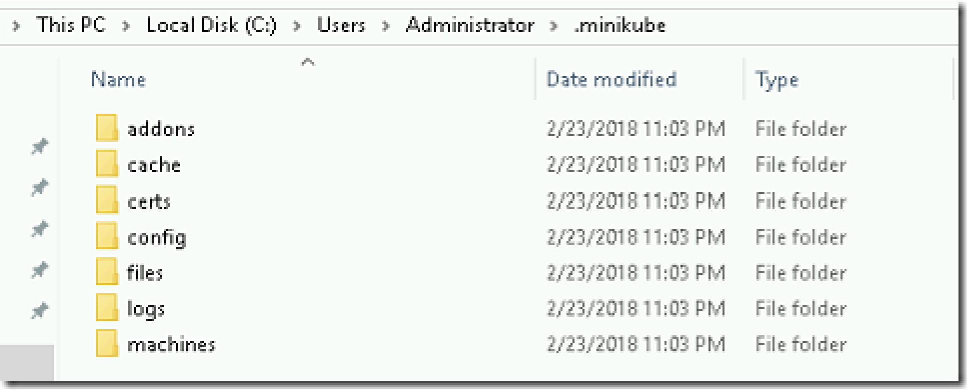
```
kubectl get pod
```



```
PS C:\zproj\blog\Groves\106CouchbaseKubernetesAzure\src> kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
cb-example-0000                     1/1     Running   0          8m
cb-example-0001                     1/1     Running   0          6m
cb-example-0002                     1/1     Running   0          3m
couchbase-operator-55f8cff68c-47ff6 1/1     Running   0          24m
PS C:\zproj\blog\Groves\106CouchbaseKubernetesAzure\src> _
```

Q20. Where is the minikube config file location?

Ans. The location for config file is: ~/.minikube/config/config.json



› This PC › Local Disk (C:) › Users › Administrator › .minikube

Name	Date modified	Type
addons	2/23/2018 11:03 PM	File folder
cache	2/23/2018 11:03 PM	File folder
certs	2/23/2018 11:03 PM	File folder
config	2/23/2018 11:03 PM	File folder
files	2/23/2018 11:03 PM	File folder
logs	2/23/2018 11:03 PM	File folder
machines	2/23/2018 11:03 PM	File folder

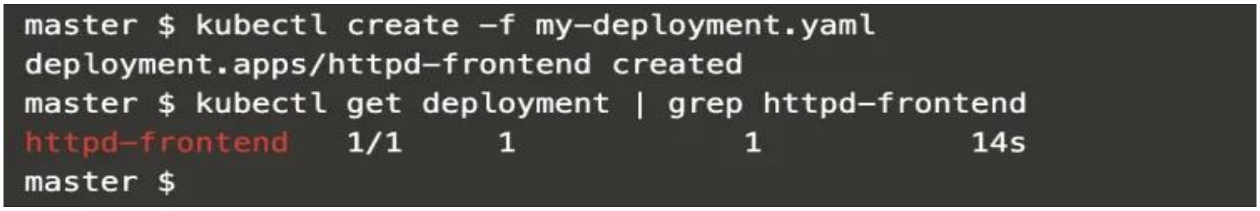
Q21. How to start minikube on a specific network?

Ans. you can use `minikube start` command with below argument

```
--host-only-cidr string (only supported with VirtualBox driver) (default "192.168.99.1/24")
```

Q22. How to create a deployment in minikube?

Ans. A deployment can be created using the kubectl run, kubectl apply, or kubectl create commands. Once created, a Deployment ensures that the desired number of Pods are running and available.



```
master $ kubectl create -f my-deployment.yaml
deployment.apps/httpd-frontend created
master $ kubectl get deployment | grep httpd-frontend
httpd-frontend   1/1     1            1           14s
master $
```

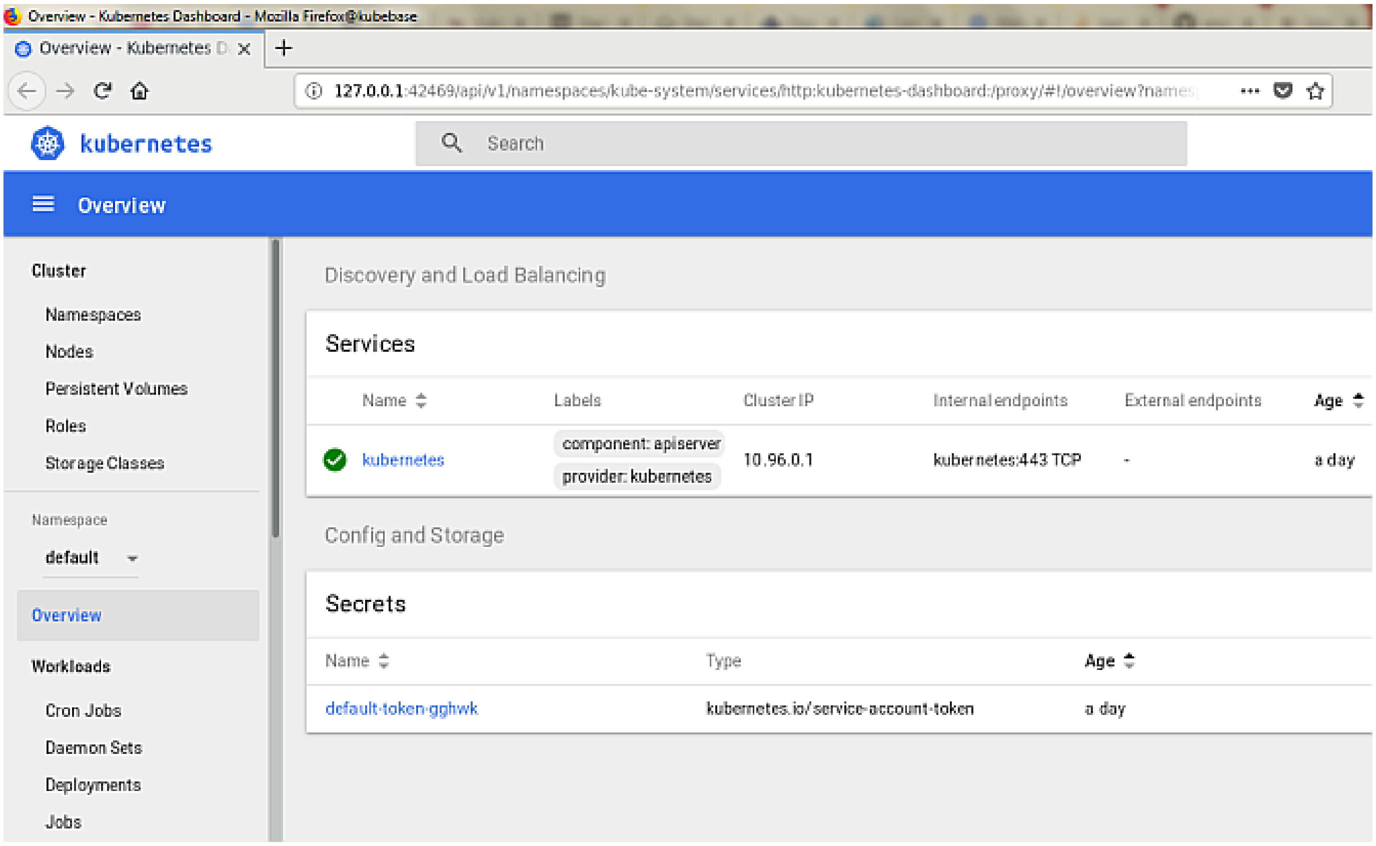
Q23. Is the minikube support dashboard?

Ans. yes, to access the dashboard:

```
minikube dashboard
```

```
[root@kubebase ~]# minikube dashboard
* Enabling dashboard ...
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:34375/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/ in your default browser...
START /usr/bin/firefox "http://127.0.0.1:34375/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/"
```

```
Running without a11y support!
```



Discovery and Load Balancing

Services

Name	Labels	Cluster IP	Internal endpoints	External endpoints	Age
✓ kubernetes	component: apiserver provider: kubernetes	10.96.0.1	kubernetes:443 TCP	-	a day

Config and Storage

Secrets

Name	Type	Age
default-token-gghwk	kubernetes.io/service-account-token	a day

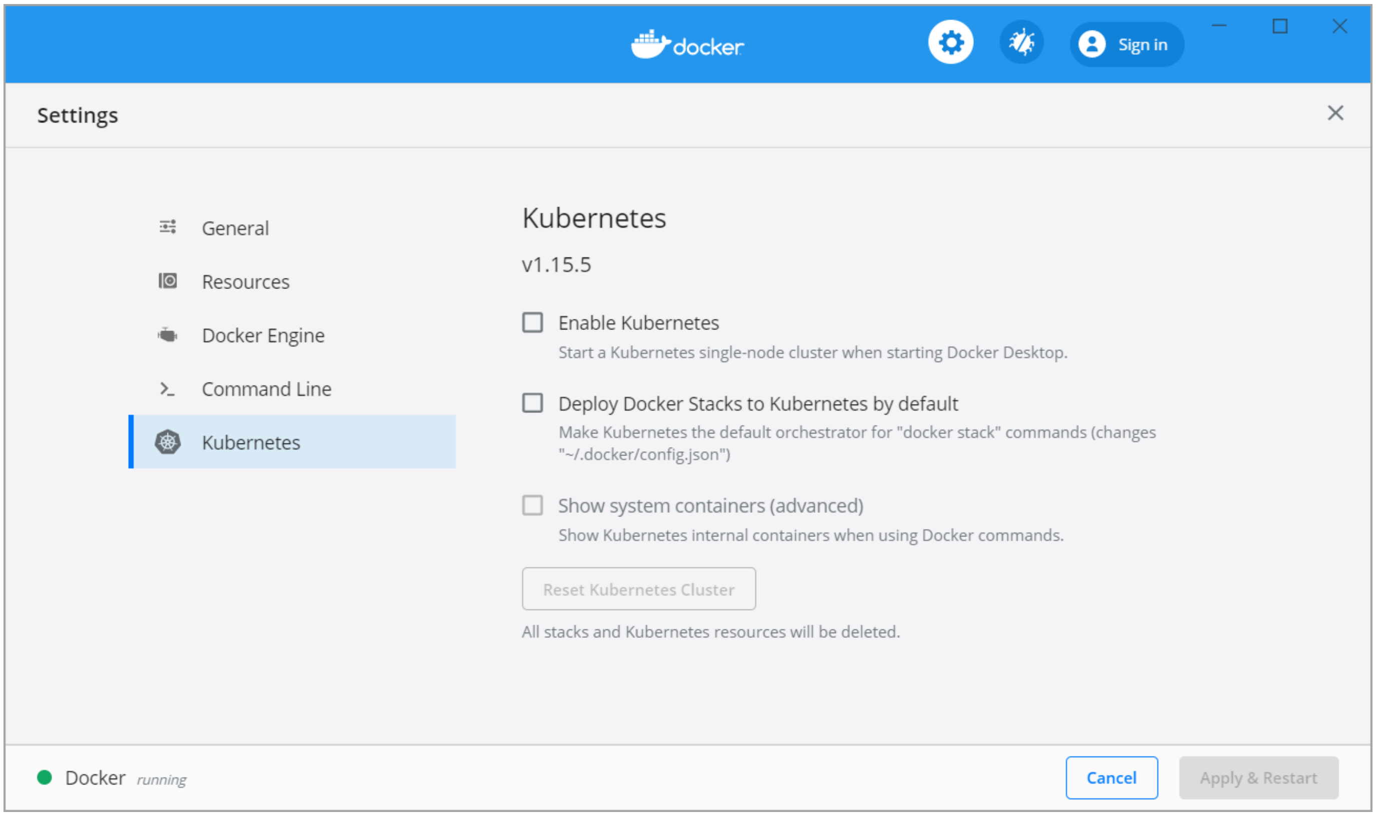
Q24. What does Kubectl stand for?

Ans. "ctl" stands for control. also, other different meaning of kubectl: “kube control”, “kube cuddle”, “kube c-t-l”, or “kubie cuttle”

Q25. Is Docker Desktop support Kubernetes?

Ans. Yes, docker desktop for windows and mac supports Kubernetes features.

Note: you can't use the Kubernetes feature when you have windows container mode.



Settings

General
Resources
Docker Engine
Command Line
Kubernetes

Kubernetes

v1.15.5

□ Enable Kubernetes
Start a Kubernetes single-node cluster when starting Docker Desktop.

□ Deploy Docker Stacks to Kubernetes by default
Make Kubernetes the default orchestrator for "docker stack" commands (changes "~/.docker/config.json")

□ Show system containers (advanced)
Show Kubernetes internal containers when using Docker commands.

Reset Kubernetes Cluster

All stacks and Kubernetes resources will be deleted.

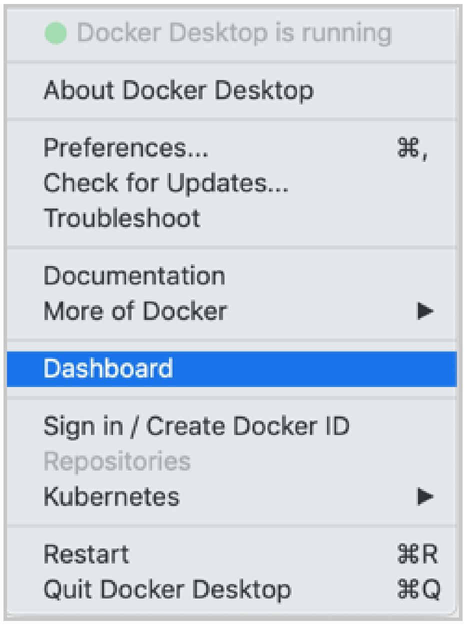
● Docker *running*

Cancel   Apply & Restart

Q26. Can we use the dashboard feature in Docker Desktop?

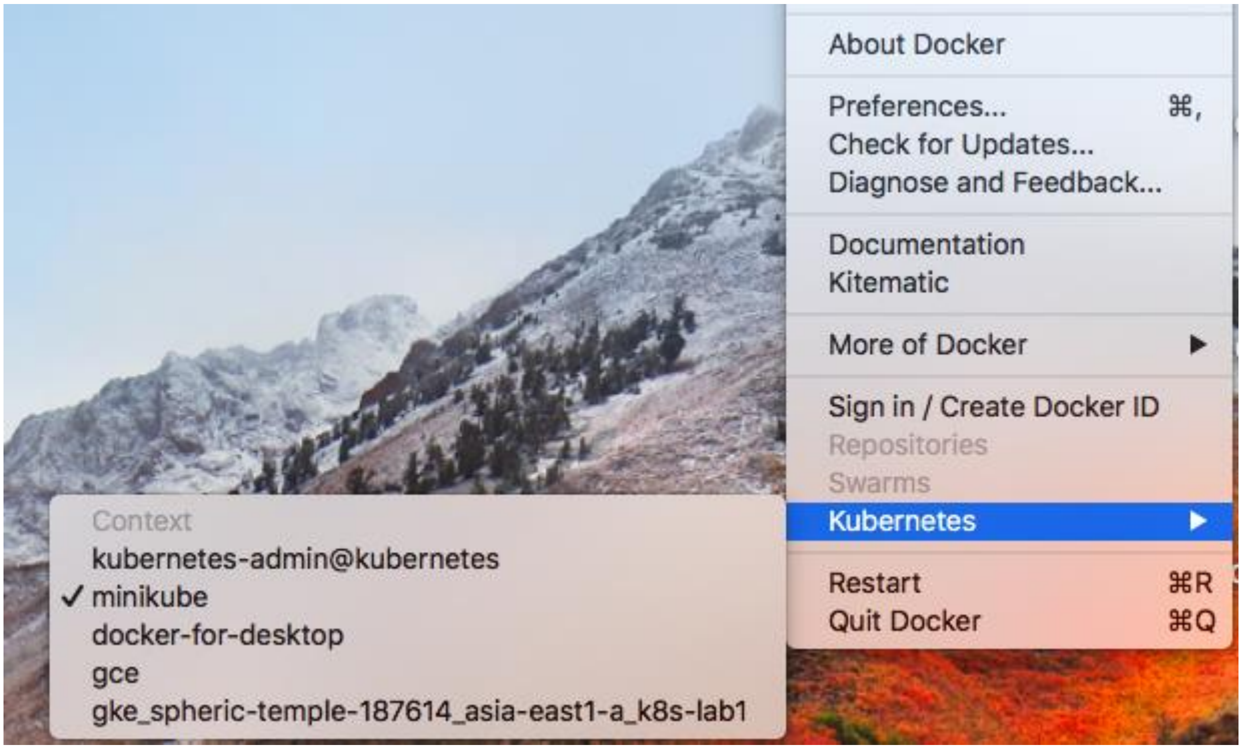
Ans. Yes, we can use the dashboard feature in Docker Desktop.using that feature you can manage your container from dashboard UI.also you can check the log, inspect your container, check the status of the container.

You can launch your dashboard from below :



Q27. How can we apply Minikube in Docker Desktop?

Ans. Once you have installed minikube in your system you can use minikube with docker desktop. In kubernetes context selection pane you can see minikube option like below:

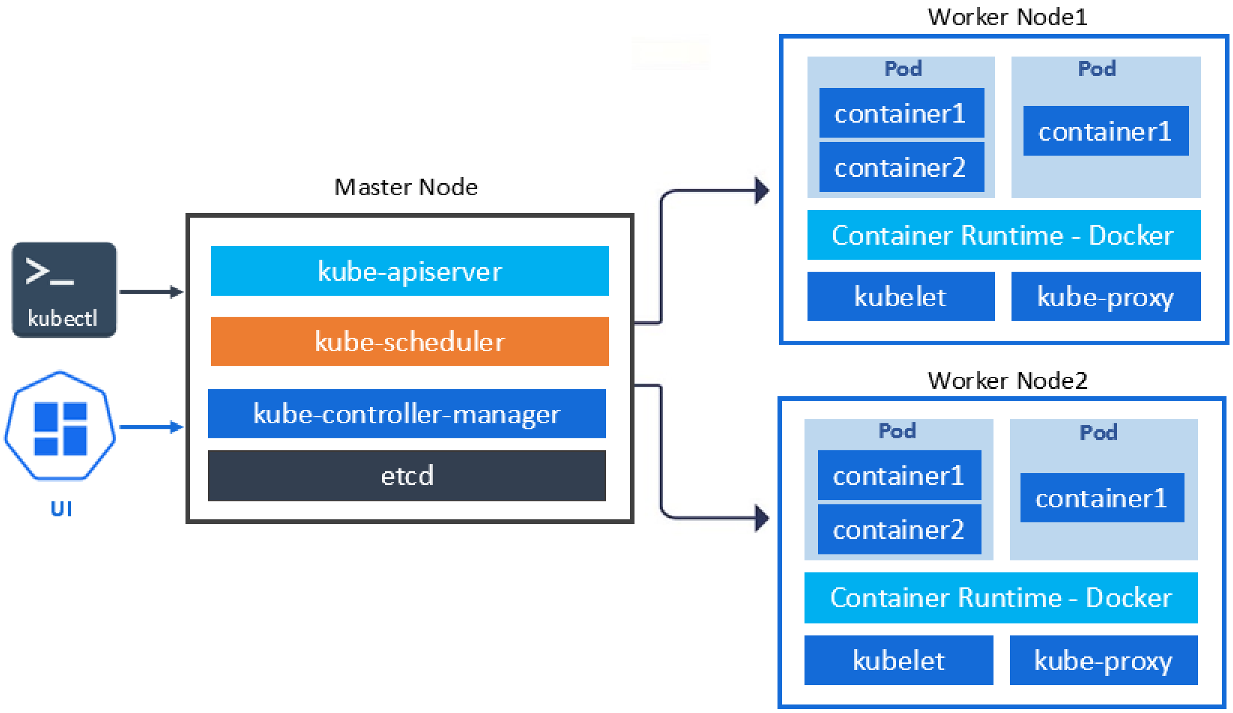


4

Kubernetes Architecture

Q1. Do explain Kubernetes Architecture?

Ans. Kubernetes Architecture has worker node and master node with some functionality like master node consist managing API and scheduling the pods and, on the worker, node runs on container runtime such as Docker, rkt along with the node that communicates with the master.



From a high level, a Kubernetes environment consists of a control plane (master), a distributed storage system for keeping the cluster state consistent (etcd), and many cluster nodes (Kubelets).

1. **Master Components**
   - Kube-apiserver
   - Etcd
   - Kube-scheduler
   - Kube-controller-manager

2. **Node Components**

- Kubelet
- Kube-Proxy
- Container Runtime

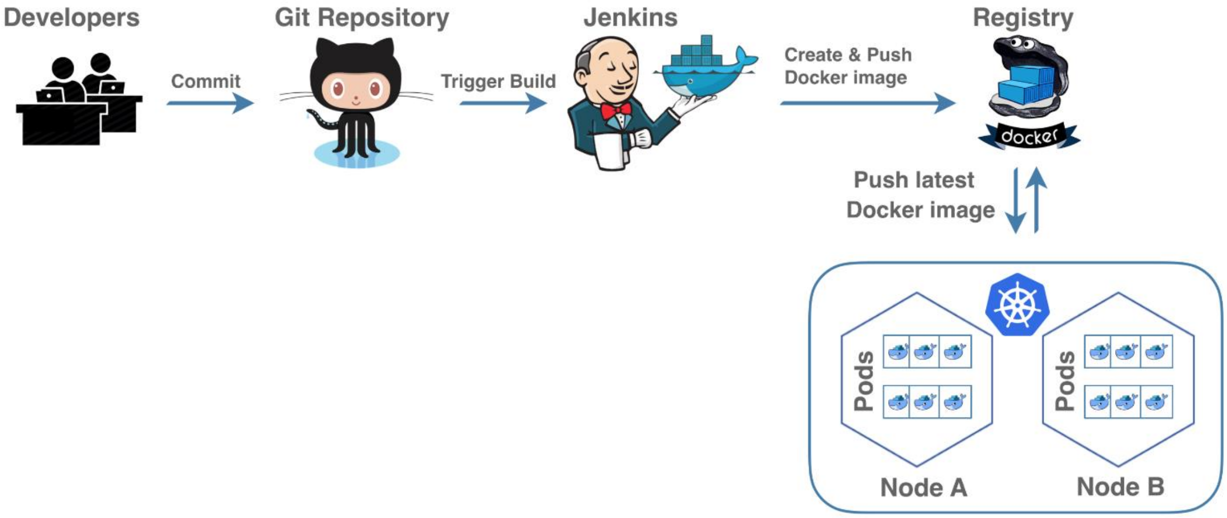
Q2. How does Kubernetes work?

Ans. Kubernetes keeps track of your container applications that are deployed into your cluster. It restarts orphaned containers, shuts down containers when they're not being used, and automatically provisions resources like memory, storage, and CPU when necessary.

Q3. How does the CI/CD pipeline work with Kubernetes?

Ans. CI/CD stands for Continuous integration - continuous delivery. When Developer pushes the code in SCM, the pipeline will trigger and perform some task like:

- Build Image
- Test Execution
- Push image to the repository
- Pull the latest image
- Run image (Up the container)



Q4. How does Kubernetes Autoscaling work?

Ans. Scaling is an essential function that used to be done manually for a long-time concerning the application, with the introduction of tools like Kubernetes, the things have changed dramatically in the IT industry. In the Kubernetes cluster, there are typically two things you would like to scale as a user, Pods, and Nodes.

There are three types of scaling:

- HorizontalPodAutoscaler
- VerticalPodAutoscaler
- Cluster Autoscaler.

HorizontalPodAutoscaler - Increasing the number of Pods for the application, in response to the present computational needs.

VerticalPodAutoscaler - Expanding the resources of the Pods.

Cluster Autoscaler (CA)- Scales node clusters based on the pending pods.

Q5. Explain Kubernetes Deployment strategies.

Ans. Need to determine which Kubernetes deployment method is best to continuously roll out updates that won't affect your users.

- **Recreate**: Destroy the old version and up the new version.
- **Rolling update**: The version b is slowly rolled out replacing version a.
- **Blue/green**: The new version is shipped alongside to old version and the traffic its switch off.
- **Canary release**: New version is deployed to a subset of users and gradually incremented for all users.
- **A/B release**: As a canary release, the subset it’s defined by specific conditions.
- **Shadowing**: The traffic is sending to both versions, and version b doesn't impact on the response.

Q6. What is Etcd?

Ans. Etcd is a store for the configuration, state, and metadata of Kubernetes clusters. Also, we can say it is a distributed key-value store. It is written in Go programming language and represents the state of the entire cluster. Nodes will refer to the global configuration data stored there to set themselves up whenever they are regenerated.

Q7. What is a Master?

Ans. A Master is the central control point that provides a view of the cluster. There is a single master node that controls multiple worker nodes. It is responsible to manage the desired state of the Kubernetes cluster, when you communicate with the Kubernetes cluster using kubectl then it will go through the Kubernetes master. We can say it's the brain of the cluster.

Q8. What is a Worker node?

Ans. A worker node is nothing but one machine with container runtime configuration, Master node assigned work and worker node actually perform the assigned tasks and report back to the Master.

Q9. What are the different objects in Kubernetes?

Ans. There are following objects in Kubernetes as:

- Pods
- Replication sets and controllers
- Deployments
- Distinctive identities
- Stateful sets
- Daemon sets
- Jobs and Cron jobs

Q10. Explain the Load balancer in Kubernetes?

Ans. Kubernetes load balancing is the distribution of load among different nodes using different methods. The load balancer uses the Kubernetes Endpoints API to track the availability of pods. Which manages the virtual IPs used by services.

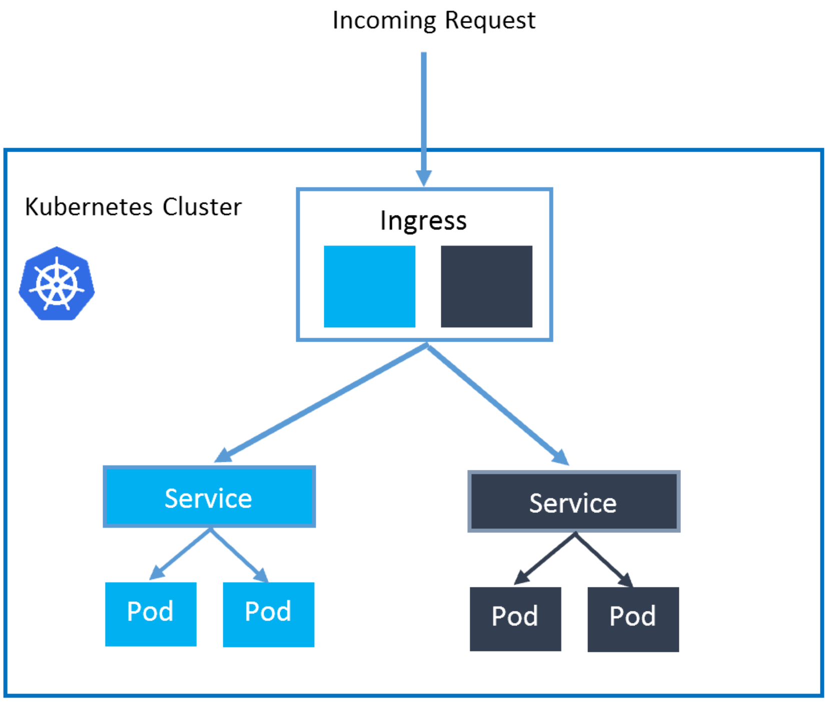
Q11. What is Ingress in Kubernetes?

Ans. In Kubernetes, Ingress is one of the core concepts that control how external users access services running in a Kubernetes cluster.Configure access by creating a set of rules that define which inbound connections reach which services.

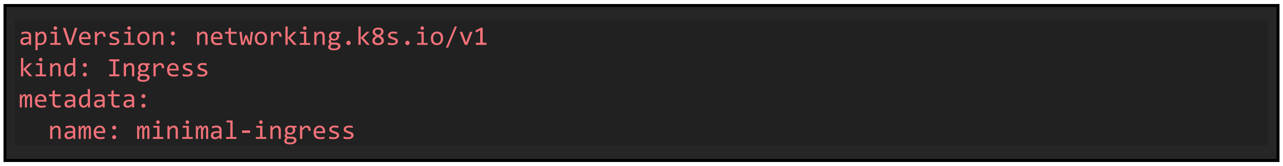
It mainly helps in implementing rules to control traffic routes. Ingress is a set up to provide services to externally reachable URLs, load balance traffic, offer name-based virtual hosting, terminate SSL (secure sockets layer) or TLS (transport layer security), etc. It is very important to note that ingress doesn't expose all ports, only HTTP and HTTPS.

The three types of ingress are:

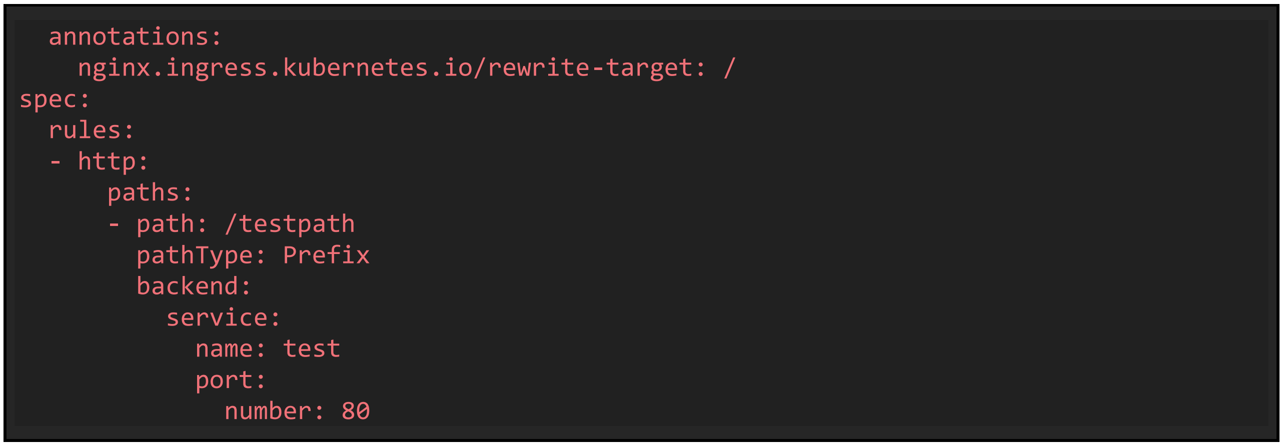
- Single service exposes a single service Ex: NodePort.
- Simple routes traffic from a single IP to multiple services based on the URI.
- Name-based virtual hosting routes traffic to multiple hostnames on the same IP.



Here Is the Ingress resource example:



```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
```



```
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /testpath
        pathType: Prefix
        backend:
          service:
            name: test
            port:
              number: 80
```

Q12. What is NodePort?

Ans. A NodePort is an open port on every node of your cluster. It is the most preferable way to get external traffic directly to your resource. It opens a specific port on all Nodes and forwards any traffic sent to this port to the service even if your application is running on a different node. Cluster-configured NodePort ranges (typically 30000–32767).

The YAML for a NodePort service looks like this:

```
apiVersion: v1
kind: Service
metadata:
 name: my-nodeport-service
spec:
 selector:
   app: my-app
 type: NodePort
 ports:
 - name: http
   port: 80
   targetPort: 80
   nodePort: 30036
   protocol: TCP
```

Q13. What is the Cluster IP?

Ans. If We want to expose service to another node within the cluster then we can achieve this using clusterIP.

Here is the yml file example:

```
apiVersion: v1
kind: Service
metadata:
 name: my-service
spec:
 selector:
   app: my-app
 type: ClusterIP
 ports:
 - name: http
   port: 80
   targetPort: 80
   protocol: TCP
```

Q14. Why do we need Kubernetes High Availability (HA)?

Ans. Depending on a single master cluster it is not a good idea because it can easily fail when something goes wrong in dependency. A multi-master cluster uses multiple master nodes, each of which has access to the same worker nodes in the cluster. Kubernetes High-Availability is about setting up Kubernetes for no single point of failure.

Q15. Is Kubernetes a container runtime?

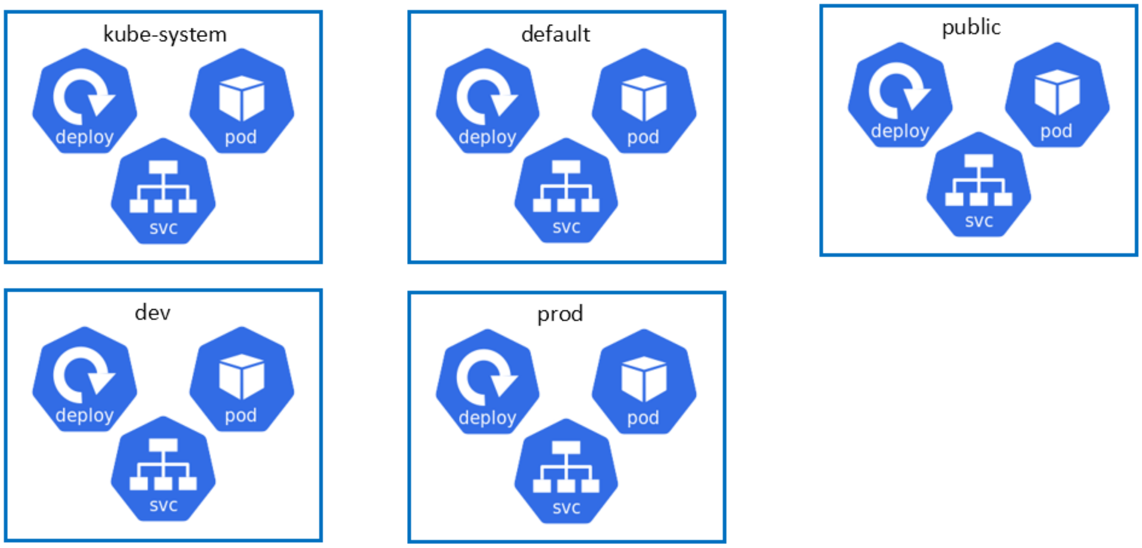
Ans. No, Kubernetes is used for container orchestration. it supports various container runtimes like docker, rkt etc.

5

Kubernetes Object & Workload

Q1. What is Namespace?

Ans. Using Namespaces If Multiple teams can use the same cluster. Ultimately namespace can create a virtual wall between Kubernetes resources.



Q2. What are the different services within Kubernetes?

Ans. There are the following services in Kubernetes:

- Cluster IP service
- Node Port service
- External Name Creation service and
- Load Balancer service

Q3. What are the built-in namespaces available in Kubernetes?

Ans. The built-In namespaces are:

- Default
- Kube - system
- Kube – public

Q4. Explain Daemon sets?

Ans. A Daemon Set is a set of pods that are run only once on a host, It creates pods on each added node and garbage collects pods when nodes are removed from the cluster.

Q5. Explain Replica set?

Ans. Replica Set ensures how many replicas of pods should be running at any given time. It can be considered as a replacement for a replication controller.

Q6. What is Heapster?

Ans. Heapster is a performance monitoring and metrics collection tool supported natively on the Kubernetes cluster compatible with Kubernetes versions 1.0. 6 and above.

It runs like any other pod in the cluster. It discovering a collection of not only performance metrics about your workloads, pods, and containers but also events and other signals generated by your cluster

Q7. Explain Replication controllers?

Ans. Replication Controller is one of the key features of Kubernetes, which is playing the supervisor role for long-running pods. It ensures that the specified number of pods are running at the run-time and also ensures that a pod or a set of pods are alike.

Q8. What are the types of controller managers?

Ans. Types of controller Manager:

- Endpoints controller
- Namespace controller
- Service account controller
- Replication controller
- Node controller
- Token controller

Q9. What are Kubernetes labels?

Ans. A label is a meaningful tag that is attached to Kubernetes objects. It contains key-value pairs.

Q10. Why does the kubelet manage the hosts file?

Ans. Kublet is managing the host file because it prevents modification once the container starts. If you make manual changes while running containers in the host file , those changes are lost once the container exists.

Q11. How can we get a static IP for a Kubernetes load balancer?

Ans. A static IP for the Kubernetes load balancer can get by changing DNS records because the Kubernetes Master can delegate a new static IP address.

6

Kubernetes Master Components

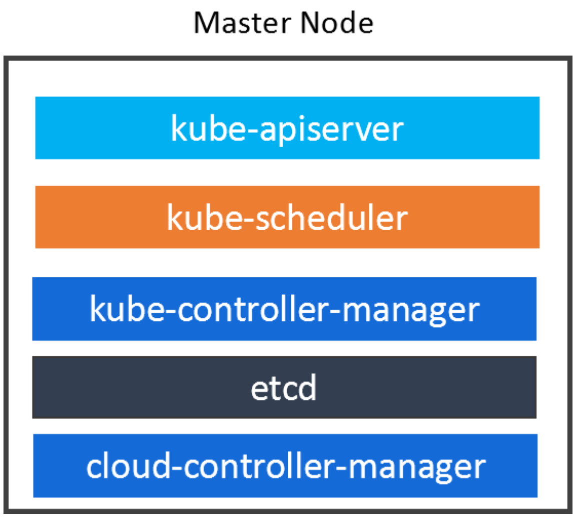
Q1. Which component can manage a Kubernetes cluster?

Ans. Kubernetes master component is responsible for managing Kubernetes clusters.

Q2. What are the main components of the master node?

Ans. Master components provide the cluster's control plane. Master components make global decisions about the cluster.

- **kube-apiserver**: exposes the Kubernetes API from master nodes
- **kube-schedule**r: schedule pods to run on selected nodes
- **kube-controller-manager**: node controller, replication controller, endpoints controller, and service account & token controllers
- **Etcd**: reliable data store for all k8s cluster data
- **cloud-controller-manager:** It allows to run Kubernetes on public, private, and hybrid clouds.



Q3. What process runs on Kubernetes Master Node?

Ans. In the master node, a different process is running to manage the entire cluster. The process like kube-api server process, scheduling, controller, and managing ETCD component process.

Q4. What is the API in Kubernetes?

Ans. The API server is the gateway to the Kubernetes cluster. It is the entrance point for all REST commands, and its associated result will be stored in etcd (or other storage). That is accessed by all users, automation, and components in the Kubernetes cluster.

Q5. How do I connect to Kubernetes API?

Ans. To access kubernetes api you need kubernetes command-line tool, kubectl. For accessing clusters you should have cluster location and credentials.

Q6. How do I run Kube API Server?

Ans. kube-apiserver is running as a container on your master node .so, its binary is binding within the container, not on your host system. It is started by the master's kubelet from a file located at `/etc/kubernetes/manifests`

Q7. Where are the stored Credentials of the API server?

Ans. The location is “`/etc/kubernetes/kubelet.conf`”

Q8. What is Kubernetes scheduler?

Ans. It schedules the pods to run on selected nodes.

Q9. How does the Kubernetes scheduler work?

Ans. In kubernetes, the Default scheduler runs as part of Master node. Scheduler will focus to assign newly created pod to the node. If the node is not appropriate for that node, the pod will remain unscheduled until the scheduler can assign.

Q10. Can we implement a custom scheduler in Kubernetes?

Ans. Yes, we can implement our custom scheduler in Kubernetes master.

Q11. What are the feasible nodes?

Ans. Nodes in a cluster that meet the scheduling requirements of a Pod are called feasible Nodes for the Pod.

Q12. How does the kube scheduler do the selection of nodes?

Ans. kube-scheduler selects a node for the pod in a two method:

- Filtering: In the filtering, Method finds the set of Nodes where it’s feasible to schedule the Pod
- Scoring: The scheduler ranks the remaining nodes to choose the most suitable Pod placement. The scheduler assigns a score to each Node that survived filtering, basing this score on the active scoring rules.

Q13. How can we check the kube scheduler log in master node?

Ans. For checking the log, you have to go in this file: /var/log/kube-scheduler.log

Q14. What is a kube-controller manager?

Ans. In kubernetes kube-controller is the main component of the cluster. It manages different control processes like nodes, workloads, namespace, service accounts and many more in the cluster. Also control loop watches the live status of the cluster using API server and manage the current state and desired state of the container.

Q15. What is Etcd?

Ans. The brain of the Kubernetes cluster.

- Etcd is the storage for configuration and secrets.
- Ideally, Kubernetes cluster run on multiple machines, it is one type of distributed system so it is required distributed datastore like ETCD
- Anything you read from a 'kubectl get ABC' command is all stored in etcd.
- Any changes made via kubectl create will create an entry in etcd to be updated.
- Any info about the node crashing or process dying makes values in etcd to be changed.
- This is how the set of processes that make up Kubernetes use etcd, to store data and notify each other of any changes that happened over time.

Q16. What is cloud-controller?

Ans. cloud-controller-manager is a daemon acting like an abstraction layer between the APIs and the different cloud providers and it is responsible for managing controller processes with dependencies on the underlying cloud provider. cloud-controller-manager is handled if a node was terminated or set up routes, load balancers or volumes in the cloud infrastructure etc.

Q17. What is the job of the kube-scheduler?

Ans. Responsibility of the kube scheduler to assign newly created pods to respective nodes.

Q18. What happens if Kubernetes master goes down?

Ans. Whenever the master node under Kubernetes fails, API will be offline, the cluster remains in an operational mode.

7

Kubernetes Node Components

Q1. What is a node in Kubernetes?

Ans. A node is one machine where containers will run. It represents a single component of a machine in a cluster, which could be a physical machine or a virtual machine. Each machine can communicate with any other machine in a Kubernetes cluster. The master in Kubernetes controls the nodes.

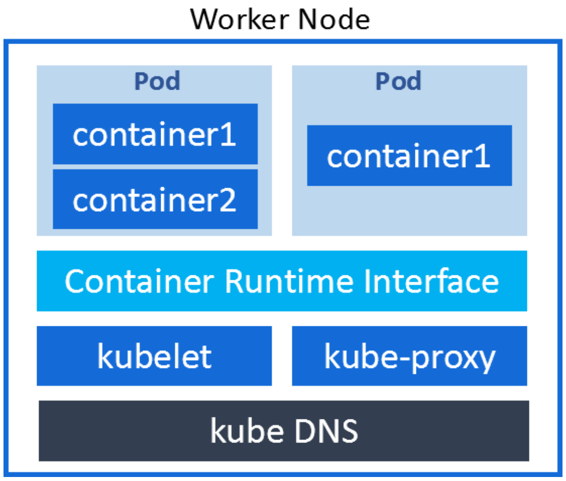
Q2. Explain Kubernetes node components?

Ans. It is known as the worker machine in the cluster (minion)

- This machine performs the requested tasks. The Master Node controls each Node
- Runs containers inside pods
- This is the place where Docker engines run and allow images to download and start containers as per requested

The node components are:

- **Kubelet**: makes sure that containers are running in a pod
- **Kube-proxy**: perform connection forwarding
- **Container Runtime Interface (CRI)**: Kubernetes supported runtimes: Docker, rkt, runc and any OCI runtime-spec implementation.
- **Kube DNS**: allow pods to communicate with each other using their name or FQDN (Fully Qualified Domain Name) instead of their local IP.



Q3. What is kubelet?

Ans. Kubelet is:

- A service which runs on each node and master in the cluster.
- Send the details to the kube API server.
- Midway between the kube-apiserver and the CRI (Container Runtime Interface)
- Gets the pod specifications from kube-apiserver and ensures that pods and their containers are healthy and running in the desired state.
- A service, not a POD.
- Uses the container runtime to start the pod, monitors its lifecycle and checks the status etc.

Q4. What is the kube-proxy in Kubernetes?

Ans. Kube proxy is one service which is run on every node in a cluster.It is a managed networking rule of the entire cluster.using network proxy, pode will communicate inside and outside of your cluster.

Q5. What task is Kube Proxy responsible for?

Ans. Main responsibility to rout request(traffic) to the appropriate container based on IP address and assign port of the incoming request.

Q6. What is Container Runtime Interface (CRI)?

Ans. To create containers on nodes we need at least one container runtime interface on that node. You can use different container runtimes like Docker, containerd, cri-o, rktlet, but widely used is docker.

Q7. How does DNS work in Kubernetes?

Ans. The Domain Name System (DNS) is a system for associating various types of information – such as IP addresses – with easy-to-remember names. DNS is a built-in service and provides a lightweight mechanism for service discovery in Kubernetes.It will configure automatically when you configure Kubernetes cluster.

Kubernetes Domain Name Server schedules a DNS Pod and Service on the cluster, and set up the kubelet to inform individual containers to use the DNS Service’s IP to resolve DNS names.

Q8. What are the major functions of Kubelet as a node service component in Kubernetes?

Ans. Major operations of kubelet as node service is run the pod containers and Report the status of the node and each pod to the API Server

Q9. What things do we need to take care before joining the node in k8s cluster?

Ans. Here is the list of steps:

- check your Kubernetes cluster is working – configured and working control plane node
- Make sure you have installed Container runtime (Docker, cri-o, containerd, e.t.c) and Kubernetes tools (kubeadm and kubelet) in your Worker node.

- If using Firewall such as firewalld, ports 10250, 30000-32767 and ports required by your Pod network add-on should be opened in the firewall.
- SSH access to the machine to be added
- Configured kubectl for checking if the node is available in your cluster

Q10. How can we join Kubernetes nodes to existing clusters?

Ans. You need to follow this step:

❖ Get join Token

- Run the command on Control node

```
kubeadm token create --print-join-command
```

❖ Get Discovery Token CA cert Hash

- Run the command on Control node

```
openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -
pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed
's/^.* //'
```

❖ Get API Server Advertise address

```
kubectl cluster-info
```

Output:

```
Kubernetes master is running at https://192.168.101.21:6443

KubeDNS is running at https://192.168.101.21:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

Metrics-server is running at https://192.168.101.21:6443/api/v1/namespaces/kube-system/services/https:metrics-server:/proxy
```

- **Note**: As shown in the output, my API is on https://192.168.101.21:6443

❖ Join a new Kubernetes Worker Node a Cluster

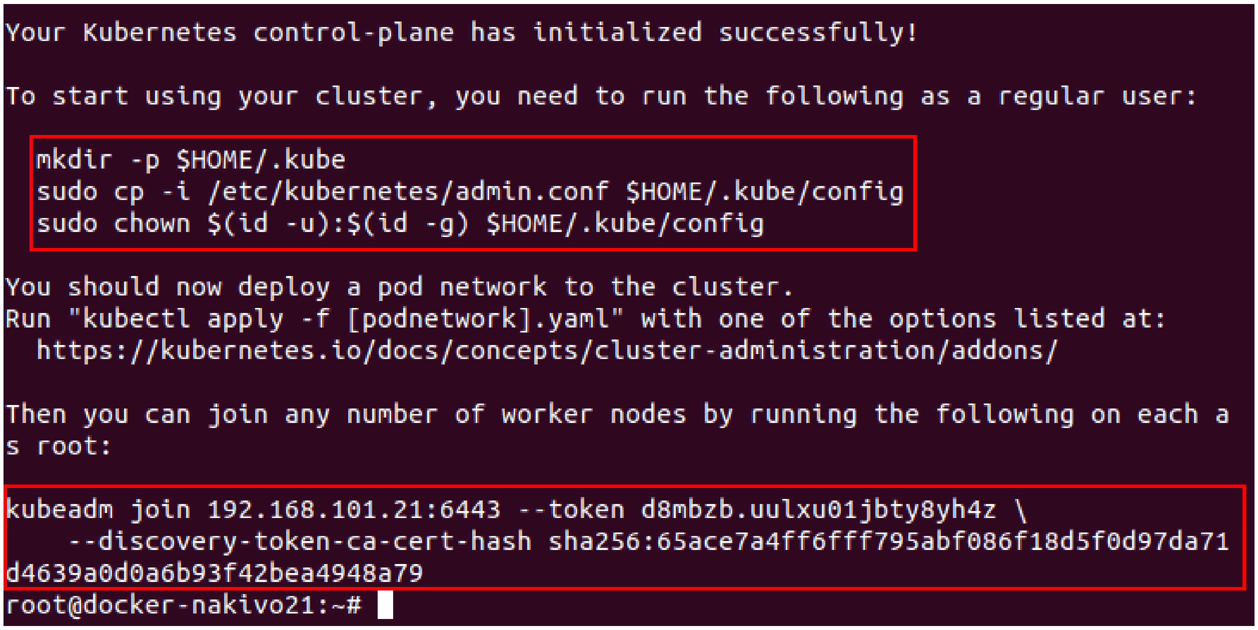
- you need to run these commands:

```
kubeadm join \

  <control-plane-host>:<control-plane-port> \

  --token <token> \

  --discovery-token-ca-cert-hash sha256:<hash>
```

- 

```
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each a
s root:

kubeadm join 192.168.101.21:6443 --token d8mbzb.uulxu01jbty8yh4z \
    --discovery-token-ca-cert-hash sha256:65ace7a4ff6fff795abf086f18d5f0d97da71
d4639a0d0a6b93f42bea4948a79
root@docker-nakivo21:~# 
```

  - ❖ Verification:
    - Run the command on control node

```
watch kubectl get nodes
```

Q11. What does the node status Hold?

Ans. The main components of a node status are Address, Condition, Capacity, and Info.

8

Kubernetes Installation

Q1. How do you Verify if Kubernetes is installed?

Ans. This can be found within the dashboard or at the UCP API endpoint version. The other way to find Kubernetes versions is from Docker CLI. You need CRI and type the following command.

```
docker --version
```

Q2. Is Kubernetes free?

Ans. Yes, Kubernetes is an open-source container-orchestration system for automating application deployment, scaling, and management.

Q3. What are the Prerequisites for Kubernetes?

Ans. Hardware Requirements:

- Ubuntu 16.04+
- Debian 9
- CentOS 7
- RHEL 7
- Fedora 25/26 (best-effort)
- HypriotOS v1.0.1+
- Container Linux (tested with 1800.6.0)

Minimal required memory & CPU (cores):

- Master node’s minimal required memory is 2GB and the worker node needs minimum is 1GB
- The master node needs at least 1.5 and the worker node needs at least 0.7 cores.

Cluster Setup:

- kubeadm: the command to bootstrap the cluster.
- kubelet: the agent that runs on all of the machines in your cluster
- kubectl: the command-line tool to talk to your cluster.
- All network connectivity access between all machines in the cluster (public or private network is fine)
- Unique hostname, MAC address, and product_uuid for every node. See here for more details.
- Certain ports are open on your machines
- Swap disabled. You MUST disable swap for the kubelet to work properly

Q4. Which ports need to open to create a k8s cluster?

Ans. The following ports need to open for k8s cluster.

Control-plane node(s)

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	6443*	Kubernetes API server	All
TCP	Inbound	2379-2380	etcd server client API	kube-apiserver, etcd
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	10251	kube-scheduler	Self
TCP	Inbound	10252	kube-controller-manager	Self

Worker node(s)

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	30000-32767	NodePort Services†	All

Q5. How to check the MAC address and product_uuid are unique for every node?

Ans. Verify Mac address: `ip link or ifconfig -a`

Verify product_uuid: `sudo cat /sys/class/dmi/id/product_uuid`

Q6. How can we assure that iptable can see bridge traffic?

Ans. We need to set this configuration for assurity:

net.bridge.bridge-nf-call-iptables is set to 1 in your sysctl config

For Example:

```
cat <<EOF > /etc/sysctl.d/k8s.conf

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

EOF

sysctl --system
```

Q7. Why do we need Check network adapters when we are going to install k8s?

Ans. If we have more than one network adapter, and Kubernetes components are not reachable on the default route, for that we need to add IP route(s) so Kubernetes cluster addresses go via the appropriate adapter.

Q8. What is used by Kubernetes to run containers inside the pod?

Ans. Kubernetes uses a container runtime (Docker)to run containers inside the pod.

Kubernetes supported runtimes: Docker, rkt, runc and any OCI runtime-spec implementation

Q9. How to install Kubernetes in Linux?

Ans. Follow below instruction:

Full fill all prerequisites

OS which are supported:

- Ubuntu 16.04+
- Debian 9+
- CentOS 7
- Red Hat Enterprise Linux (RHEL) 7
- Fedora 25+
- HypriotOS v1.0.1+
- Container Linux (tested with 1800.6.0)

Hardware/Cluster:

- 2 GB or more of RAM per machine (any less will leave little room for your apps)
- 2 CPUs or more
- Full network connectivity between all machines in the cluster (public or private network is fine)
- Unique hostname, MAC address, and product_uuid for every node
- Certain ports are open on your machines.
- Swap disabled. You MUST disable swap for the kubelet to work properly

Verify the MAC address and product_uuid

- Verify Mac address: ip link or ifconfig -a
- Verify product_uuid: sudo cat /sys/class/dmi/id/product_uuid

Check Network Adapter

Recommend you add IP route(s) so Kubernetes cluster addresses go via the appropriate adapter

Letting iptables see bridged traffic

We should ensure net.bridge.bridge-nf-call-iptables is set to 1 in your sysctl config, e.g.

```
cat <<EOF > /etc/sysctl.d/k8s.conf

net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1

EOF
```

sysctl --system

Make sure that the br_netfilter module is loaded before this step. This can be done by running lsmod | grep br_netfilter. To load it explicitly call modprobe br_netfilter

Check required ports

Control-plane node(s)

Protocol	**Direction**	**Port Range**	**Purpose**	**Used By**
TCP	Inbound	6443*	Kubernetes API server	All
TCP	Inbound	2379-2380	etcd server client API	kube-apiserver, etcd
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	10251	kube-scheduler	Self
TCP	Inbound	10252	kube-controller-manager	Self

Worker node(s)

Protocol	Direction	Port Range	Purpose	Used By
TCP	Inbound	10250	Kubelet API	Self, Control plane
TCP	Inbound	30000-32767	NodePort Services†	All

Installing Container runtime

you need to install docker as container runtime

Kubernetes uses the Container Runtime Interface (CRI) to interface with your chosen container runtime.

If we don't specify the container runtime, kubeadm automatically tries to detect an installed container runtime by scanning through a list of well-known Unix domain sockets. The following table lists container runtimes and their associated socket paths:

Runtime Path to Unix domain socket

Docker	/var/run/docker.sock
containerd	/run/containerd/containerd.sock
CRI-O	/var/run/crio/crio.sock

If both Docker and containerd are detected, Docker takes precedence

Installing kubeadm, kubelet and kubectl

kubeadm will not install or manage kubelet or kubectl

you will need to ensure they match the version,For example, kubelets running 1.7.0 should be fully compatible with a 1.8.0 API server, but not vice versa.

You must use a kubectl version that is within one minor version difference of your cluster.

Install CNI plugins (required for most pod network):

```
CNI_VERSION="v0.8.2"
```

```
mkdir -p /opt/cni/bin
```

```
curl-L "https://github.com/containernetworking/plugins/releases/download/${CNI_VERSION}/cni-plugins-linux-amd64-${CNI_VERSION}.tgz" | tar -C /opt/cni/bin -xz
```

Install crictl (required for Container Runtime Interface (CRI))

```
CRICTL_VERSION="v1.17.0"
```

```
mkdir -p /opt/bin
```

```
curl-L "https://github.com/kubernetes-sigs/cri-tools/releases/download/${CRICTL_VERSION}/crictl-${CRICTL_VERSION}-linux-amd64.tar.gz" | tar -C /opt/bin -xz
```

Install kubeadm, kubelet, kubectl and add the service:

```
RELEASE="$(curl -sSL https://dl.k8s.io/release/stable.txt)"
```

```
mkdir -p /opt/bin
```

```
cd /opt/bin
```

```
curl -L --remote-name-all https://storage.googleapis.com/kubernetes-release/release/${RELEASE}/bin/linux/amd64/{kubeadm,kubelet,kubectl}
```

```
chmod +x {kubeadm,kubelet,kubectl}
```

```
RELEASE_VERSION="v0.2.7"
```

```
curl -sSL "https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/kubepkg/templates/latest/deb/kubelet/lib/systemd/system/kubelet.service" | sed "s:/usr/bin:/opt/bin:g" > /etc/systemd/system/kubelet.service
```

```
mkdir -p /etc/systemd/system/kubelet.service.d
```

```
curl -sSL "https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/kubepkg/templates/latest/deb/kubeadm/10-kubeadm.conf" | sed "s:/usr/bin:/opt/bin:g" > /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```

Enable and start kubelet:

```
systemctl enable --now kubelet
```

Configure cgroup driver used by kubelet on control-plane node

For Docker, kubeadm will automatically detect the cgroup driver and it will set at runtime on below location:

/var/lib/kubelet/kubeadm-flags.env

For different CRI, you have to modify the file /etc/default/kubelet (/etc/sysconfig/kubelet for CentOS, RHEL, Fedora) with your cgroup-driver value,like:

```
KUBELET_EXTRA_ARGS=--cgroup-driver=<value>
```

Restarting the kubelet is required:

```
systemctl daemon-reload
```

```
systemctl restart kubelet
```

Q10. How to install kubectl in windows?

Ans. Steps:

- Download the latest binary: “ Link.”
- To find the latest version refer this link
- “https://storage.googleapis.com/kubernetes-release/release/stable.txt”
- Add the binary into your PATH.
- Test to ensure the version of kubectl: kubectl version --client

Q11. How to install kubelet on MacOS?

Ans. Steps:

1. **Download the latest release:**

```
curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/darwin/amd64/kubectl"
```

2. **For a specific version, replace the**

$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt) portion of the command with the specific version. For example, to download version v1.18.0 on macOS, type:

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.18.0/bin/darwin/amd64/kubectl
```

3. **Make the kubectl binary executable.**

```
chmod +x ./kubectl
```

4. **Move the binary into your PATH.**

```
sudo mv ./kubectl /usr/local/bin/kubectl
```

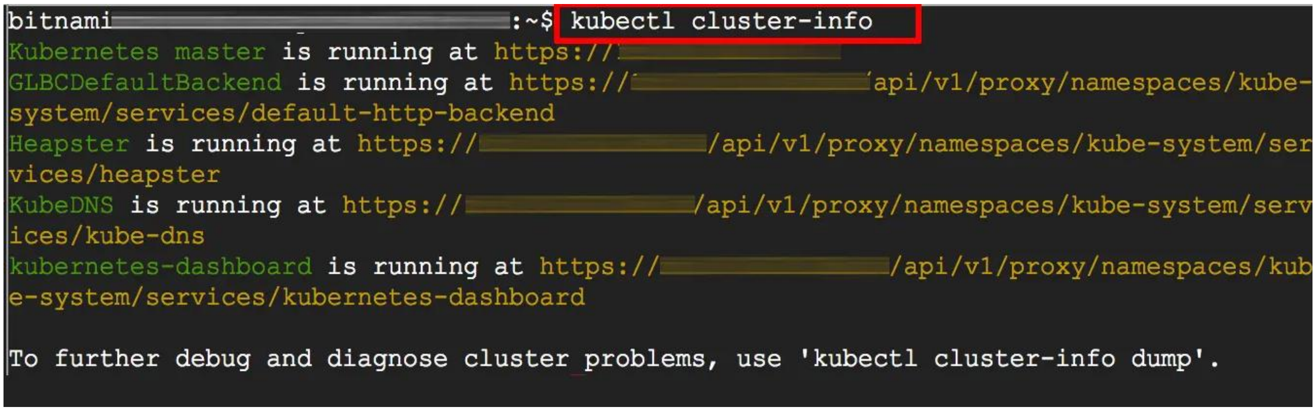
5. **Verify Installation:**

```
kubectl version --client
```

Q12. How to check if a Kubernetes cluster is running or not?

Ans. You can use the following commands:

kubectl cluster-info



```
bitnami[redacted]:~$ kubectl cluster-info
Kubernetes master is running at https://[redacted]
GLBCDefaultBackend is running at https://[redacted]api/v1/proxy/namespaces/kube-
system/services/default-http-backend
Heapster is running at https://[redacted]/api/v1/proxy/namespaces/kube-system/ser
vices/heapster
KubeDNS is running at https://[redacted]/api/v1/proxy/namespaces/kube-system/serv
ices/kube-dns
kubernetes-dashboard is running at https://[redacted]/api/v1/proxy/namespaces/kub
e-system/services/kubernetes-dashboard

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

Q13. How to update Kubernetes cluster?

Ans. *kubectl apply* is a command that will update a Kubernetes cluster to match state defined locally in files

Q14. How to Restart the kubelet?

Ans. Below commands need to perform:

systemctl daemon-reload

systemctl restart kubelet

9

Kubernetes Commands

Q1. How to List pods with nodes info?

Ans.
```
kubectl get pod -o wide
```

Q2. What is the command for List everything in Kubernetes cluster?

Ans.
```
kubectl get all --all-namespaces
```

Q3. How can we Get all the services?

Ans.
```
kubectl get service --all-namespaces
```

Q4. What is the command for Get all deployments?

Ans.
```
kubectl get deployments --all-namespaces
```

Q5. How can we Show nodes with labels?

Ans.
```
kubectl get nodes --show-labels
```

Q6. Can you tell the commands for Get resources with json output?

Ans.
```
kubectl get pods --all-namespaces -o json
```

Q7. How to Validate a yaml file with dry run?

Ans.
```
kubectl create --dry-run --validate -f pod.yaml
```

Q8. How to Get system config via configmap?

Ans.
```
kubectl -n kube-system get cm kubeadm-config -o yaml
```

Q9. What is the command for Watching pods?

Ans.
```
kubectl get pods -n wordpress --watch
```

Q10. What is the Query health check endpoint?

Ans.
```
curl -L http://127.0.0.1:10250/healthz
```

Q11. How can we Open a bash terminal in a pod?

Ans.

```
kubectl exec -it storage sh
```

Q12. How can we check the environment variable of the pod?

Ans.

```
kubectl exec my-pod env
```

Q13. How can we use the “kubectl apply command” on the folder where our yml file is located?

Ans.

```
kubectl apply -R -f .
```

Q14. How can we Get services sorted by name?

Ans.

```
kubectl get services -sort-by=.metadata.name
```

Q15. How can we Get pods sorted by restart count?

Ans.

```
kubectl get pods -sort-by=’.status.containerStatuses[0].restartCount’
```

Q16. How can we get a list of pods and images?

Ans.

```
kubectl get pods
-o=’custom-columns=PODS:.metadata.name,Images:.spec.containers[*].image’
```

Q17. How can we list out all container images?

Ans.

```
kubectl get pods --all-namespaces -o jsonpath="{..image}" |\
tr -s '[[:space:]]' '\n' |\
sort |\
uniq -c
```

Q18. How can we Get node resource usage?

Ans.

```
kubectl top node
```

Q19. How can we Get pod resource usage?

Ans.

```
kubectl top pod
```

Q20. How to List resource utilization for all containers?

Ans.

```
kubectl top pod --all-namespaces --containers=true
```

Q21. What is the command for Delete pod?

Ans.

```
kubectl delete pod/<pod-name> -n <my-namespace>
```

Q22. What is the command for Delete pod by force?

Ans.

```
kubectl delete pod/<pod-name> --grace-period=0 --force
```

Q23. How can we Delete pods by labels?

Ans.

```
kubectl delete pod -l name=myLabel
```

Q24. How can we Delete all resources filtered by labels?

Ans.

```
kubectl delete pods,services -l name=myLabel
```

Q25. How can we Delete persistent volumes by labels?

Ans.

```
kubectl delete pvc -l name=myLabel
```

Q26. What is the command for List all critical pods?

Ans.

```
kubectl get -n kube-system pods -a
```

Q27. How can we List Out pods with more info?

Ans.

```
kubectl get pod -o wide,

kubectl get pod/<pod-name> -o yaml
```

Q28. How can we get pod info?

Ans.

```
kubectl describe pod/<pod_name>
```

Q29. How can we List all pods with labels?

Ans.

```
kubectl get pods --show-labels
```

Q30. How can we List all unhealthy pods?

Ans.

```
kubectl get pods -field-selector=status.phase!=Running -all-namespaces
```

Q31. How can we list out only running pods?

Ans.

```
kubectl get pods -field-selector=status.phase=Running
```

Q32. How can we Filter pods by label?

Ans.

```
kubectl get pods -l owner=mike
```

Q33. How can we Manually add labels to a pod?

Ans.

```
kubectl label pods <pod-name> owner=denny
```

Q34. What is the command for removing labels of pods?

Ans.

```
kubectl label pods dummy-input owner-
```

Q35. How can we List secrets in Kubernetes?

Ans.
```
kubectl get secrets --all-namespaces
```

Q36. How can we Generate secret?

Ans.
```
echo -n 'mypasswd', then redirect to base64 --decode
```

Q37. How can we Get secret?

Ans.
```
kubectl get secret denny-cluster-kubeconfig
```

Q38. How can we List all Secrets currently in use by a pod?

Ans.
```
kubectl get pods -o json | jq '.items[].spec.containers[].env[]?.valueFrom.secretKeyRef.name' | grep -v null | sort | uniq
```

Q39. How can we Get a specific field of a secret?

Ans.
```
kubectl get secret denny-cluster-kubeconfig -o jsonpath=”{.data.value}”
```

Q40. How can we List storage class?

Ans.
```
kubectl get storageclass
```

Q41. How can we Check the mounted volumes?

Ans.
```
kubectl exec storage ls /data
```

Q42. How can we Check persistent volume?

Ans.
```
kubectl describe pv/pv0001
```

Q43. How can we Copy local file to the pod?

Ans.
```
kubectl cp /tmp/my <some-namespace>/<some-pod>:/tmp/server
```

Q44. How can we Copy pod file to local?

Ans.
```
kubectl cp <some-namespace>/<some-pod>:/tmp/server /tmp/my
```

Q45. How can we View all events?

Ans.
```
kubectl get events --all-namespaces
```

Q46. How can we List Events sorted by timestamp?

Ans.
```
kubectl get events –sort-by=.metadata.creationTimestamp
```

Q47. How can we Mark node as unschedulable?

Ans.
```
kubectl cordon <node-name>
```

Q48. How can we Mark node as schedulable?

Ans.

```
kubectl uncordon <node-name>
```

Q49. How can we Drain node in preparation for maintenance?

Ans.

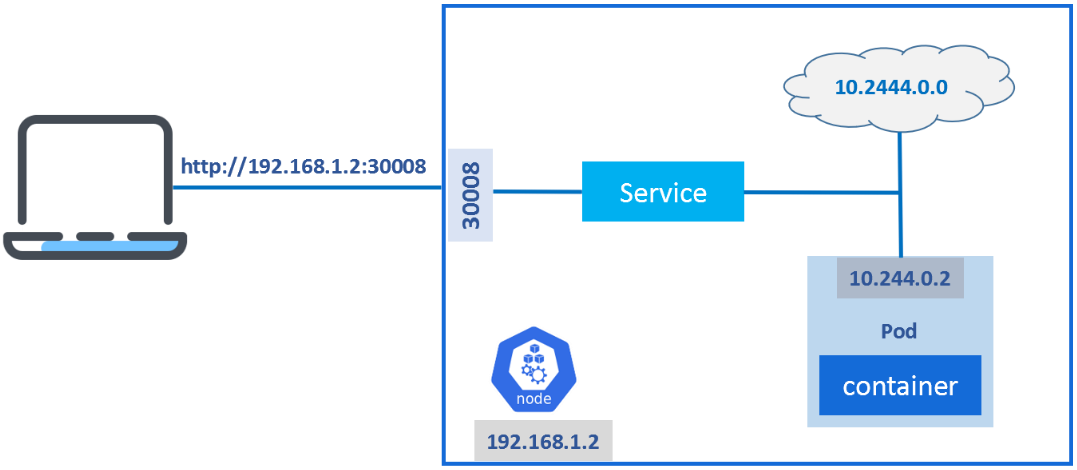
```
kubectl drain <node-name>
```

10

Kubernetes Services

Q1. What is the Kubernetes service?

Ans. A kubernetes service is a REST object that provides a single point of access from the outside world, into your pod(s) which run your application.



Q2. How does Kubernetes service work?

Ans. Services are programs that manage the list of the pods IP for a deployment. Services select Pods based on their labels. When a network request comes to the service, it selects all Pods matching the service's selector, chooses one of them, and serves the network request back.

Q3. What is the difference between deployment and service Kubernetes?

Ans. A **deployment** is responsible to run a set of pods in a cluster. To run a set of identical pods in a cluster you can use the only deployment without service configuration. The deployment could be scaled up and down and pods could be replicated. Each pod could be accessed individually via direct requests of accessing pods.

A **service** is responsible for enabling network access to a set of pods. Service is responsible to expose an application as network service. To use service without deployment, you need to create pods individually rather than "all-at-once”. Then service requests traffic rout the network requests based on their pod's labels.

Services and Deployments are totally different, but they work together nicely.

Q4. How do services connect to the deployment?

Ans. Using selectors you can connect service and deployment. Keep in mind, in Kubernetes it depends on the YML file to check the system is running as according to the desired state to the YML file.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
 name: nginx-deployment
spec:
 replicas: 2
 template:
  metadata:
   labels:
    app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:1.17.0
```

services.yml file

```
kind: Service
metadata:
 name: frontend
spec:
 type: NodePort
 selector:
    app: nginx
 ports:
  - port: 80
    targetPort: 80
    nodePort: 30005
```

Q5. What are the different types of services in Kubernetes?

Ans. There are four types of service in Kubernetes:

- **ClusterIP** - is one type of service and only other pods can access that service within the same cluster.
- **NodePort** - is an open port for every node within the same cluster so incoming traffic can route to your service even if it is for a different node's application.
- **LoadBalancer** - can access the service from outside the Kubernetes cluster (using an external IP). Load Balancers are great when you have a TCP level connection. for higher-level connection (http) use Ingress + NodePort.
- **ExternalName** - can access the service to the contents of the externalName field (e.g., fn.bam.example.com)

Kube-proxy implements a form of virtual IP for services for all types other than ExternalName. To achieve this, you can set three possible modes:

- Proxy-mode: userspace
- Proxy-mode: iptables

- Proxy-mode: ipvs

Q6. How to Identify a Kubernetes service?

Ans. There are two ways to identify service.

- DNS
- ENV Var

Q7. How do I access Kubernetes service?

Ans. We can access service using different ways:

- Access services through public IPs
- Access services, nodes, or pods using the Proxy Verb
- Access from a node or pod in the cluster

Q8. What is the Kubernetes headless service?

Ans. Sometimes you don’t need load-balancing and a single service IP. In this case, you can create what is termed “headless” Services A headless service is a service with a service IP but instead of load-balancing, it will return the IPs of our associated Pods. This allows us to communicate directly with the Pods instead of a proxy.

Explicitly specifying "**None**" for the cluster IP **(.spec.clusterIP).**

Q9. What is the default protocol for a Service?

Ans. TCP is the default protocol for service.

Q10. What is the command for Listing all Kubernetes services?

Ans.

```
kubectl get services
```

Q11. What is the command for listing service endpoints?

Ans.

```
kubectl get endpoints
```

Q12. How can we Get service details?

Ans.

```
kubectl get service <service-name> -o yaml
```

Q13. How can we Get service cluster IP?

Ans.

```
kubectl get service <service-name> -o go-template='{{.spec.clusterIP}}’
```

Q14. How can we get service cluster port?

Ans.

```
kubectl get service <service-name> -o go-template='{{(index .spec.ports 0).port}}’.
```

Q15. How can we Expose service as load balancer service?

Ans.

```
kubectl expose service/wordpress-1-svc --type=LoadBalancer --name=my-service
```

11

Kubernetes Pods

Q1. What is a pod and why are pods so important?

Ans. Generally, it refers to one or more containers in a pod. Pods are handling Volumes, Secrets, and configuration for containers. Pods are short-lived. They are planned to be restarted automatically when they die.

A pod can be created by using YAML syntax as given below:

```
kubectl apply -f pod.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    name: myapp
spec:
  containers:
    - name: nginx-container
      image: nginx
```

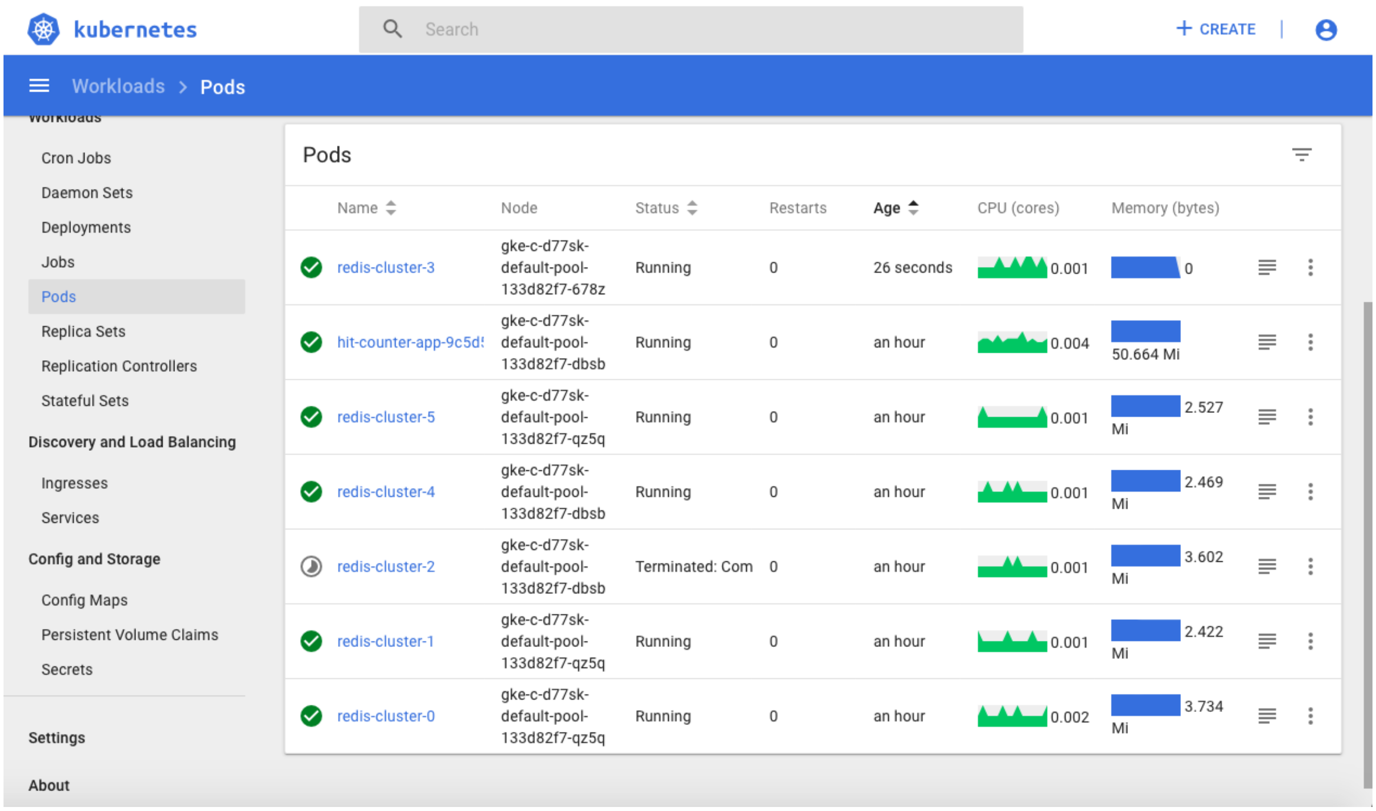
Pods are replicated when the app is scaled horizontally by the ReplicationSet.Pods are the most basic and one of the foremost units in Kubernetes. Other cluster objects are here to serve them and all the tasks they do are to manage pods' desired state.

Q2. Does kubelet create pods?

Ans. kubelet does not create the pod itself!

Q3. How to check if a pod is running from the dashboard?

Ans. You can go to your dashboard to check and manage your application pods.



Pods

	Name	Node	Status	Restarts	Age	CPU (cores)	Memory (bytes)
✓	redis-cluster-3	gke-c-d77sk-default-pool-133d82f7-678z	Running	0	26 seconds	0.001	0
✓	hit-counter-app-9c5d!	gke-c-d77sk-default-pool-133d82f7-dbsb	Running	0	an hour	0.004	50.664 Mi
✓	redis-cluster-5	gke-c-d77sk-default-pool-133d82f7-qz5q	Running	0	an hour	0.001	2.527 Mi
✓	redis-cluster-4	gke-c-d77sk-default-pool-133d82f7-dbsb	Running	0	an hour	0.001	2.469 Mi
◔	redis-cluster-2	gke-c-d77sk-default-pool-133d82f7-dbsb	Terminated: Com	0	an hour	0.001	3.602 Mi
✓	redis-cluster-1	gke-c-d77sk-default-pool-133d82f7-qz5q	Running	0	an hour	0.001	2.422 Mi
✓	redis-cluster-0	gke-c-d77sk-default-pool-133d82f7-qz5q	Running	0	an hour	0.002	3.734 Mi

Q4. How do I start a pod in Kubernetes?

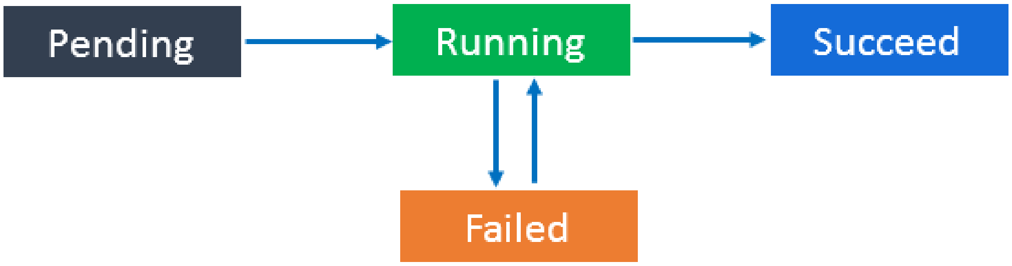
Ans. You can start pod by running the following commands:

```
kubectl apply -f myapp.yaml
```

Q5. Do explain pod lifecycle?

Ans. Through its lifecycle, a Pod can attain the following states:

- **Pending**: Pod has been accepted by the Kubernetes system, but one or more container images have not yet been created.
- **Running**: In this state, pod has been bound to respective nodes and a container has been created inside the pod. At least one container is running or is in a startup or restart state.
- **Successful** (Succeeded): All containers in the Pod are successfully terminated and will not be restarted.
- **Failed**: All containers in the Pod have been terminated, among them at least one container was terminated or exited with non-zero status in the pod.
- **CrashLoopBackoff**: The container fails to start and is tried again and again. Unable to obtain the status of the Pod for some reason, usually because of the communication with the host where the Pod is located fails.



Q6. When and how pods will be created?

Ans. When you run the command like `kubectl create -f my app.yml` The API will store this configuration in the ETCD, and the Scheduler will assign your application pods to worker nodes.

Q7. Why do we need pods in Kubernetes?

Ans. We need pods in clusters because Pods enable data sharing and communication among their constituents.so we can access pod applications via the same IP and port.

Q8. How do you get pods in a pod?

Ans. use the following command.

```
kubectl get pods [pod-name-here] -n [namespace] -o
jsonpath='{.spec.containers[*].name}*
```

Q9. How do I delete the POD forcefully in Kubernetes?

Ans. Run the following command:

```
kubectl delete pod <PODNAME> --grace-period=0 --force --namespace <NAMESPACE>
```

Q10. How do containers within a pod communicate?

Ans. Containers inside a Pod can communicate with one another via localhost.

Q11. How do I access Kubernetes pod from outside?

Ans. We have different options for connecting to nodes, pods and services from outside the cluster: like public IPs.

Q12. Can pods in different namespaces communicate?

Ans. Yes, you can put the Pods behind Services and use Service DNS for communication.

- **Same namespace**: Calls to service-name allow Pods in the same namespace to communicate.
- **Different Namespace**: Calls to service-name.namespace allow Pods in different namespaces to communicate

Q13. What do containers share inside a pod?

Ans. Every container in Pod shares the network namespace, including the IP address and network ports. Containers inside a Pod can communicate with one another using localhost.

Q14. What are the different types of multiple-container pods?

Ans. Three different types of multi-container pods.

- **Sidecar**: The Sidecar pattern is used as a single node pattern to enhance the functionality of the main application. It contains the logic of the application and it sends the logic files to the registry.
- **Adapter**: It is used to standardize and normalize the output application or monitor data for aggregation. It performs restructuring, reformatting and output correction.
- **Ambassador**: It is a proxy pattern that allows other containers with a port on the localhost.

Q15. How do I delete all pods in Kubernetes?

Ans. Run the following commands:

```
kubectl delete --all pods --namespace=foo
```

Q16. Can we recover pods after deletion?

Ans. No, you can't recover after deletion. pods are mortal. Once it dies or is deleted, they cannot be recovered. Its IP and the containers that were running on it will be gone; they are ephemeral.

Q17. Can we run windows and Linux containers in the same pod?

Ans. No, you can't run Windows and Linux containers in the same pod because a pod is scheduled in node and node has specific platform and architecture.

Q18. How do I update all my pods if the image changed but the tag is the same?

Ans. For that you need to set `"imagePullPolicy: "Always"`

Q19. What is the difference between the pod and the container?

Ans. Pods are the collection of containers in Kubernetes. Containers are the set of codes and binary to compile in a pod of the application. Containers can communicate with other containers in the same pod.

Q20. How to configure Vertical pod autoscaler?

Ans. In a vertical pod autoscaler you don't have to worry about CPU and memory resources. it will automatically update the values

Configure vertical pod autoscaler with below resource:

```
apiVersion: "autoscaling.k8s.io/v1beta2"
kind: VerticalPodAutoscaler
metadata:
  name: vpa
spec:
  targetRef:
    apiVersion: "apps/v1"
      kind: Deployment
    name: my-vpa

 resourcePolicy:
    containerPolicies:
      - containerName: '*'
        minAllowed:
          cpu: 10m
          memory: 50Mi
        maxAllowed:
          cpu: 1
          memory: 500Mi
        controlledResources: ["cpu", "memory"]
```

Q21. How to configure Horizontal pod autoscaler?

Ans. For autoscaling your app using Horizontal pod autoscaler. Once we create a resource, the Horizontal Pod Autoscaler starts executing the mentioned control loop against your app with the provided parameters in the below resource:

my-hpa.yml

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: myapp
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: Pods
      pods:
        metric:
          name: myapp_requests_per_second
        target:
          type: AverageValue
          averageValue: 2
```

Q22. How many pods can run on a node?

Ans. Mostly depends on the node and its capacity.

On Amazon Elastic Kubernetes Service (EKS), the maximum number of pods per node depends on the node type and ranges from 4 to 737.

On Google Kubernetes Engine (GKE), the limit is 100 pods per node, regardless of the type of node.

12

Kubernetes Deployments

Q1. What are Kubernetes Deployments?

Ans. Using Deployment, we can manage Pods updates and Replica Sets. You describe the desired state in a Deployment, A Deployment runs multiple replicas of your application and automatically replaces any instances that fail or become unresponsive. Deployments are managed by the Kubernetes Deployment controller.

Q2. What are Kubernetes manifests?

Ans. It is one configuration file which contains the Kubernetes "API object description". Configuration files which can describe the desired state of your application- written in YAML or JSON, using manifests you can create, modify and delete Kubernetes resources such as pods, deployments, services or ingresses. It is very common to define manifests in the form of .yaml files and send them to the Kubernetes API Server via commands “kubectl apply -f my-file”.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
 name: nginx-deployment
spec:
 replicas: 2
 template:
  metadata:
   labels:
    app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:1.17.0
```

Q3. How can we check if Deployment is created or not?

Ans. Need to run the following command:

```
kubectl get deployments
```

Q4. How do I rollback the Deployment?

Ans. **Rollback to the previously deployed version**

```
kubectl rollout undo deployment/$DEPLOYMENT
```

Rollback to a specific previously deployed version

```
kubectl rollout undo deployment/$DEPLOYMENT --to-revision 21
```

Q5. How to determine the status of deployment?

Ans. Use the below command:

kubectl rollout status

If the output runs, then the deployment is completed.

Q6. How can we see ReplicaSet which is created by deployment?

Ans. The below command gives you the ReplicaSet:

```
kubectl get rs
```

Q7. How can we get details of the deployment?

Ans. To see details of the deployments:

```
kubectl describe deployments
```

Q8. How can we Delete deployments by labels?

Ans.

```
kubectl delete deployment -l name=myLabel
```

Q9. How to Pause/Resume deployments?

Ans.

```
kubectl rollout pause deployment/nginx-deployment, resume
```

Q10. How to Check update history of deployment?

Ans.

```
kubectl rollout history deployment/nginx-app
```

Q11. How to Check the update status of Deployments?

Ans.

```
kubectl rollout status deployment/nginx-app
```

Q12. How to scale-out deployment?

Ans.

```
kubectl scale --replicas=3 deployment/<deployment-name>
```

Q13. How to Rollback Deployment to the previous version?

Ans.

```
kubectl rollout undo deployment/<deployment-name>
```

Q14. How can we Expose deployment as a load balancer service?

Ans.

```
kubectl expose deployment/my-app --type=LoadBalancer --name=my-service
```

13

Kubernetes Networking

Q1. What is Kubernetes networking?

Ans. Networking is a central part of Kubernetes. Kubernetes networking allows Kubernetes components like pod, container, service to communicate with each other and with other applications.

Q2. What are the different types of Kubernetes networking?

Ans. There are three different Kubernetes networking:

- Container-to-Container networking
- Pod-to-Pod networking
- Pod-to-Service networking
- Internet-to-Service networking

Q3. How does the Kubernetes Network work?

Ans. In Kubernetes, Every Pod has a unique IP. Kubernetes networking – through the network plug-in that is required to install (e.g. Calico, Flannel, Weave…) takes care of routing all requests internally between nodes to the appropriate pod. Using iptables kubernetes network can control and manage networking of the cluster.

Q4. What are the Kubernetes Networking Conditions?

Ans. Following are the Kubernetes networking conditions.

- All Pods can communicate with all other Pods without using network address translation (NAT)
- Nodes are the machines that run the Kubernetes cluster. agents on a node (e.g. system daemons, kubelet) can communicate with all pods on that node
- Each Pod will see itself with the same IP that other Pods see it as having.

Q5. Why do we need network policy in Kubernetes?

Ans. We need a network policy to control traffic flow at the IP and port level configuration. Network Policy can be implemented by using network plugins(like-cni)

The CNI plugin is selected by passing Kubelet the --network-plugin=cni command-line option

Q6. How to apply network policy on pod?

Ans. When we create a pod at a time the pod is in a non-isolated mode, menace pod is allowing all the traffic from the source. Once we apply, the NetworkPolicy pod becomes isolated so it will allow traffic according to the NetworkPolicy.

Below is network manifest file as reference:

```
kubectl apply -f NetworkPolicy.yaml
```

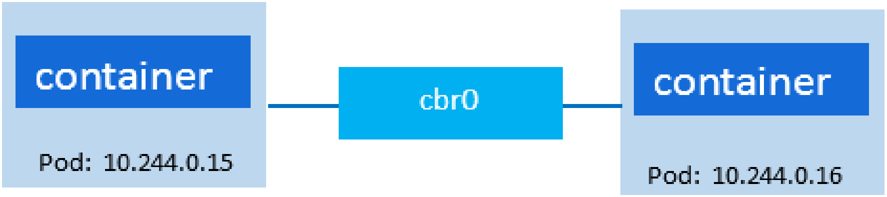
```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - ipBlock:
        cidr: 172.17.0.0/16
        except:
        - 172.17.1.0/24
    - namespaceSelector:
        matchLabels:
          project: myproject
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 6379
  egress:
  - to:
    - ipBlock:
        cidr: 10.0.0.0/24
    ports:
    - protocol: TCP
      port: 5978
```

Q7. How Does Kubernetes Networking Compare to Docker Networking?

Ans. The Kubernetes networking model natively supports multi-host networking in which pods can communicate with each other by default, regardless of which host they live in.Kubernetes manages networking through CNI's plugins.The Docker networking model relies, by default, on a virtual bridge network called Docker0. It's also important to mention that k8s does not use docker0

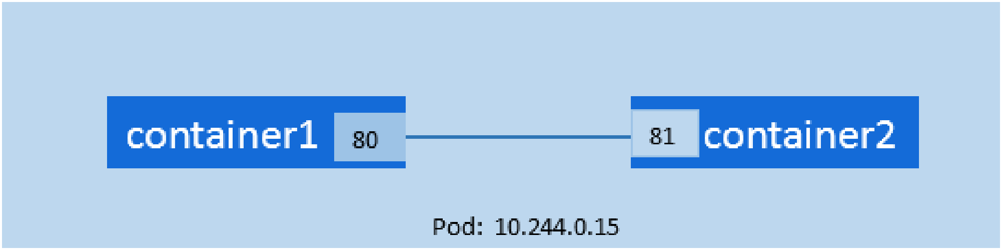
Q8. What is cbr0 in Kubernetes?

Ans: K8s does not use Docker's default bridge (docker0), but it creates its own bridge named as cbr0 for communication between pods.



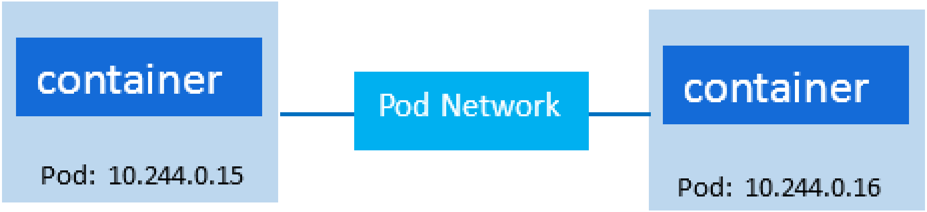
Q9. What is Container to Container network?

Ans. Using Network Namespace you can achieve container to Container networking and communication. All the containers within a given Service will have the same IP address and port space, which is assigned by Network Namespace. Network namespace has its logical router, firewall, and other network devices. They can communicate with one another via localhost.



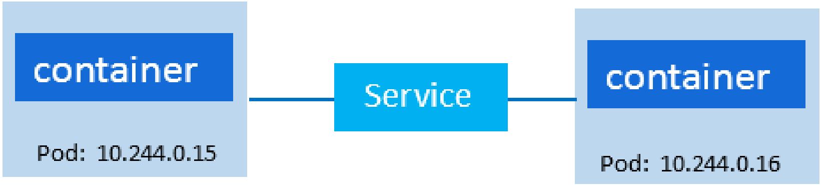
Q10. What is pod to pod network?

Ans. Pod to Pod Networking has a real IP address. Using Root network Namespace with eth0 will create a virtual interface (veth0, veth1,.) for every Pod and which is assigned to each Pod network namespace. Every virtual interface use virtual bridge to connect which will send and receive data using ARP protocol.



Q11. What is pod to service network?

Ans. Traffic will rout easily in pod with their associated until we need to deal with change. when we scaling up and down pod IP will change it is not durable that is why service has come in the picture. The set of Pods that are linked to a Service can be changed at any time, but the IP address of the Service will remain static.



Q12. Explain the Internet to Cluster Networking

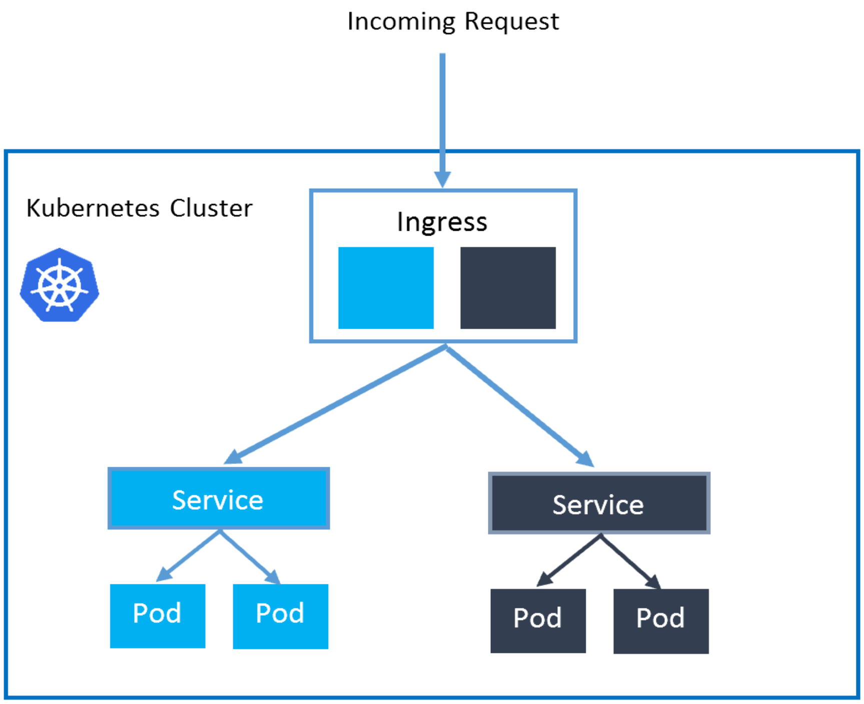
Ans. External Internet network traffic can be divided into two parts

- Ingress is the network traffic from the Internet to Kubernetes clusters.
- Egress is the network traffic from Kubernetes clusters to the Internet.

Q13. What is Ingress?

Ans. Ingress is the network execution that is getting network traffic from the internet or client to Kubernetes clusters. Internet traffic routing to Kubernetes has two different approaches:

- Load Balancer
- Ingress Controller



Q14. What is Egress?

Ans. Egress network is the way to communicate between Kubernetes cluster to the internet.

- Packet originates at the Pod's namespace.
- Connected to the root namespace via veth
- the iptables will do the NAT-ing from virtual bridge (vbr0) to internet gateway via eth0.
- At Last, the Internet Gateway will change the source IP address from internal IP to External IP so that the outer world will receive what the Kubernetes cluster sends.

Q15. What is CNI (container network interface)?

Ans. A CNI, which is short for Container Networking Interface, it simply a link between the container runtime (like Docker or rkt) and the network plugin. The Container Networking Interface (CNI) provides a common API for connecting containers to the outside network.

Q16. How many Kubernetes Network drives are supported in windows?

Ans. Windows supports five different networking drivers/modes: L2bridge, L2tunnel, Overlay, Transparent, and NAT.

Q17. What are the different types of CNI?

Ans. Most popular CNI are:

- Calico
- Canal (Flannel + Calico)
- flannel
- kopeio-vxlan
- kube-router
- romana
- Weave Net

Q18. How do I check my network for Kubernetes?

Ans. Use the kubectl get pod command on your local machine, with the option -o wide. It will give information about cluster IP.

Q19. What is a flannel network in Kubernetes?

Ans. Flannel is a virtual network that gives a subnet to each host for use with container runtimes.

References

This book has been written by referring to the following sites:

- https://kubernetes.io/docs/concepts
- https://docs.docker.com/docker-for-windows