

What is Kubernetes?

Kubernetes is also known as '**k8s**'. This word comes from the Greek language, which means a **pilot** or **helmsman**.

Kubernetes is an extensible, portable, and open-source platform designed by **Google** in **2014**. It is mainly used to automate the deployment, scaling, and operations of the container-based applications across the cluster of nodes. It is also designed for managing the services of containerized apps using different methods which provide the scalability, predictability, and high availability.

It is actually an enhanced version of '**Borg**' for managing the long-running processes and batch jobs. Nowadays, many cloud services offer a Kubernetes-based infrastructure on which it can be deployed as the platform-providing service. This technique or concept works with many container tools, like **docker**, and follows the client-server architecture.

Key Objects of Kubernetes

Following are the key objects which exist in the Kubernetes:

Pod

It is the smallest and simplest basic unit of the Kubernetes application. This object indicates the processes which are running in the cluster.

Node

A **node** is nothing but a single host, which is used to run the virtual or physical machines. A node in the Kubernetes cluster is also known as a minion.

Service

A **service** in a Kubernetes is a logical set of pods, which works together. With the help of services, users can easily manage load balancing configurations.

ReplicaSet

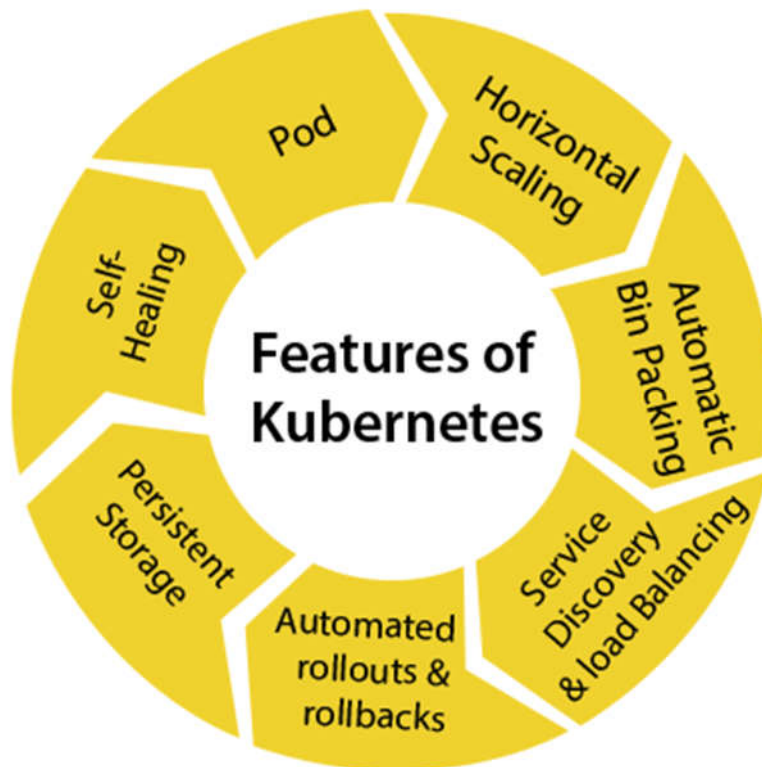
A **ReplicaSet** in the Kubernetes is used to identify the particular number of pod replicas are running at a given time. It replaces the replication controller because it is more powerful and allows a user to use the "set-based" label selector.

Namespace

Kubernetes supports various virtual clusters, which are known as namespaces. It is a way of dividing the cluster resources between two or more users.

Features of Kubernetes

Following are the essential features of Kubernetes:

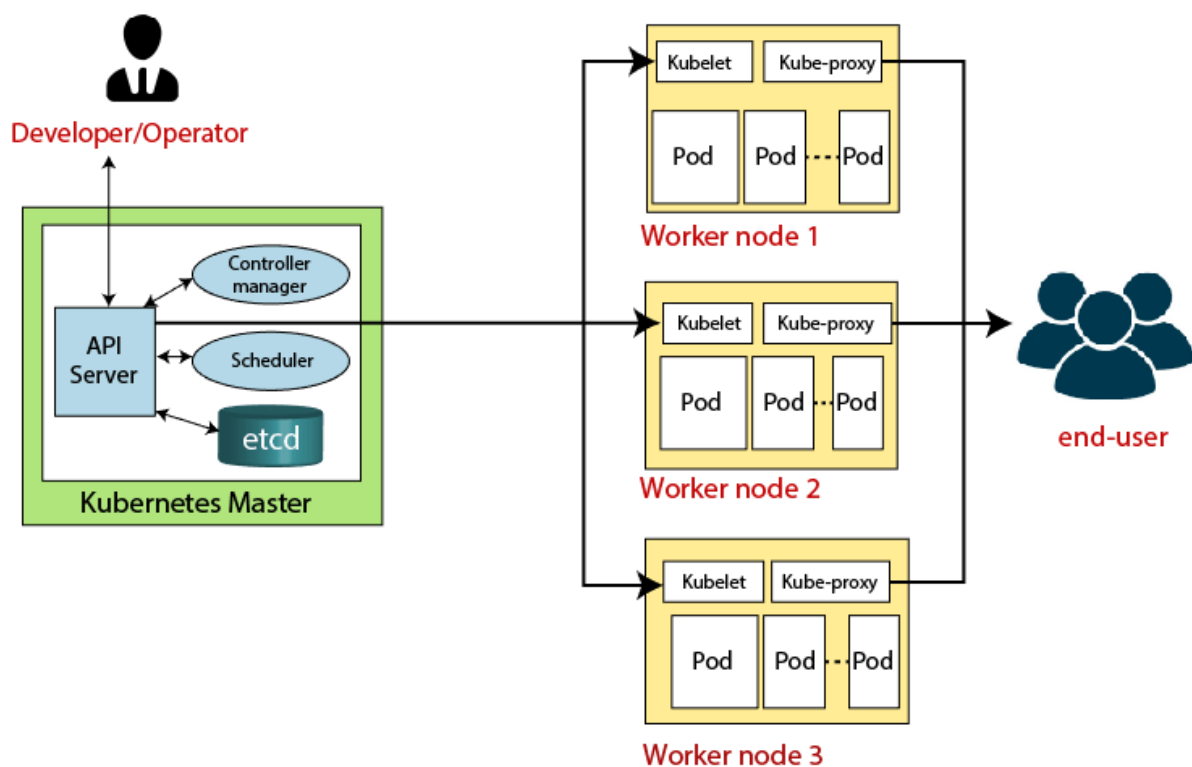


1. **Pod:** It is a deployment unit in Kubernetes with a single Internet protocol address.
2. **Horizontal Scaling:** It is an important feature in the Kubernetes. This feature uses a **HorizontalPodAutoscaler** to automatically increase or decrease the number of pods in a deployment, replication controller, replica set, or stateful set on the basis of observed CPU utilization.
3. **Automatic Bin Packing:** Kubernetes helps the user to declare the maximum and minimum resources of computers for their containers.
4. **Service Discovery and load balancing:** Kubernetes assigns the IP addresses and a Name of DNS for a set of containers, and also balances the load across them.
5. **Automated rollouts and rollbacks:** Using the rollouts, Kubernetes distributes the changes and updates to an application or its configuration. If any problem occurs in the system, then this technique rollbacks those changes for you immediately.
6. **Persistent Storage:** Kubernetes provides an essential feature called '**persistent storage**' for storing the data, which cannot be lost after the pod is killed or

rescheduled. Kubernetes supports various storage systems for storing the data, such as **Google Compute Engine's Persistent Disks (GCE PD)** or **Amazon Elastic Block Storage (EBS)**. It also provides the distributed file systems: **NFS or GFS**.

7. **Self-Healing:** This feature plays an important role in the concept of Kubernetes. Those containers which are failed during the execution process, Kubernetes restarts them automatically. And, those containers which do not reply to the user-defined health check, it stops them from working automatically.

Kubernetes Architecture



Kubernetes Architecture

The architecture of Kubernetes actually follows the client-server architecture. It consists of the following two main components:

1. Master Node (Control Plane)
2. Slave/worker node

Master Node or Kubernetes Control Plane

The master node in a Kubernetes architecture is used to manage the states of a cluster. It is actually an entry point for all types of administrative tasks. In the Kubernetes cluster, more than one master node is present for checking the fault tolerance.

Following are the four different components which exist in the Master node or Kubernetes Control plane:

1. API Server
2. Scheduler
3. Controller Manager
4. ETCD

API Server

The Kubernetes API server receives the REST commands which are sent by the user. After receiving, it validates the REST requests, process, and then executes them. After the execution of REST commands, the resulting state of a cluster is saved in '**etcd**' as a distributed key-value store.

Scheduler

The scheduler in a master node schedules the tasks to the worker nodes. And, for every worker node, it is used to store the resource usage information. In other words, it is a process that is responsible for assigning pods to the available worker nodes.

Controller Manager

The Controller manager is also known as a controller. It is a daemon that executes in the non-terminating control loops. The controllers in a master node perform a task and manage the state of the cluster. In the Kubernetes, the controller manager executes the various types of controllers for handling the nodes, endpoints, etc.

ETCD

It is an open-source, simple, distributed key-value storage which is used to store the cluster data. It is a part of a master node which is written in a GO programming language.

Now, we have learned about the functioning and components of a master node; let's see what is the function of a slave/worker node and what are its components.

Worker/Slave node

The Worker node in a Kubernetes is also known as minions. A worker node is a physical machine that executes the applications using pods. It contains all the essential services which allow a user to assign the resources to the scheduled containers.

Following are the different components which are presents in the Worker or slave node:

Kubelet

This component is an agent service that executes on each worker node in a cluster. It ensures that the pods and their containers are running smoothly. Every **kubelet** in each worker node communicates with the master node. It also starts, stops, and maintains the containers which are organized into pods directly by the master node.

Kube-proxy

It is a proxy service of Kubernetes, which is executed simply on each worker node in the cluster. The main aim of this component is request forwarding. Each node interacts with the Kubernetes services through **Kube-proxy**.

Pods

A **pod** is a combination of one or more containers which logically execute together on nodes. One worker node can easily execute multiple pods.

Installation of Kubernetes on Linux

The installation of Kubernetes on Linux is a straight forward process. Follow the below steps to install the Kubernetes. In the installation of Kubernetes, each step is mandatory.

Step 1: In this step, we have to update the necessary dependencies of a system using two commands.

The first command is used to get all the updates. Execute the following command in the terminal; it will ask to enter the system's password.

1. `sudo apt-get update`

Output:

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get update  
[sudo] password for sumit:  
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 k  
B]  
Ign:2 http://ppa.launchpad.net/couchdb/stable/ubuntu bionic InRelease  
Ign:3 http://dl.google.com/linux/chrome/deb stable InRelease  
Get:4 http://dl.google.com/linux/chrome/deb stable Release [943 B]  
Ign:5 http://ppa.launchpad.net/paolorotolo/android-studio/ubuntu bionic I  
nRelease  
Get:6 http://dl.google.com/linux/chrome/deb stable Release.gpg [819 B]  
Err:6 http://dl.google.com/linux/chrome/deb stable Release.gpg  
  
The following signatures couldn't be verified because the public key is  
not available: NO_PUBKEY 78BD65473CB3BD13  
Hit:7 http://in.archive.ubuntu.com/ubuntu bionic InRelease  
Get:8 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packag  
es [670 kB]  
Get:9 http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7  
kB]
```

When the first command is successfully executed, type the following second command, which is used to make the repositories.

1. `sudo apt-get install -y apt-transport-https`

Output:

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install -y apt-transport-https  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
apt-transport-https is already the newest version (1.6.12).  
0 upgraded, 0 newly installed, 0 to remove and 444 not upgraded.  
sumit@sumit-Vostro-15-3568:~$ sudo apt install docker.io  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
docker.io is already the newest version (19.03.6-0ubuntu1~18.04.1).  
0 upgraded, 0 newly installed, 0 to remove and 444 not upgraded.  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install docker.io  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
docker.io is already the newest version (19.03.6-0ubuntu1~18.04.1).  
0 upgraded, 0 newly installed, 0 to remove and 444 not upgraded.  
sumit@sumit-Vostro-15-3568:~$ sudo systemctl start docker  
sumit@sumit-Vostro-15-3568:~$ sudo systemctl enable docker  
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install curl  
Reading package lists... Done  
Building dependency tree
```

Step 2: After the above steps are successfully executed, we have to install the dependencies of docker in this step.

Type the following command to install the docker. In the installation process, we have to choose Y for confirmation of the installation.

1. `sudo apt install docker.io`

Output:


```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo apt install docker.io  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
docker.io is already the newest version (19.03.6-0ubuntu1~18.04.1).  
0 upgraded, 0 newly installed, 0 to remove and 444 not upgraded.  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install docker.io  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
docker.io is already the newest version (19.03.6-0ubuntu1~18.04.1).  
0 upgraded, 0 newly installed, 0 to remove and 444 not upgraded.  
sumit@sumit-Vostro-15-3568:~$ sudo systemctl start docker  
sumit@sumit-Vostro-15-3568:~$ sudo systemctl enable docker  
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service  
→ /lib/systemd/system/docker.service.  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install curl  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  libcurl4  
The following NEW packages will be installed:  
  curl libcurl4  
0 upgraded, 2 newly installed, 0 to remove and 444 not upgraded.  
Need to get 373 kB of archives.
```

After installing the docker, we have to type the different two commands for starting and enabling the docker. Type the following first command, which starts the docker:

1. `sudo systemctl start docker`

Now, type the following second command, which enables the docker:

1. `sudo systemctl enable docker`

Output:

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo systemctl start docker  
sumit@sumit-Vostro-15-3568:~$ sudo systemctl enable docker  
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service  
→ /lib/systemd/system/docker.service.  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install curl  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  libcurl4  
The following NEW packages will be installed:  
  curl libcurl4  
0 upgraded, 2 newly installed, 0 to remove and 444 not upgraded.  
Need to get 373 kB of archives.  
After this operation, 1,038 kB of additional disk space will be used.  
Do you want to continue? [Y/n] Y  
Get:1 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcu  
rl4 amd64 7.58.0-2ubuntu3.8 [214 kB]  
Get:2 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 curl  
amd64 7.58.0-2ubuntu3.8 [159 kB]  
  
Fetched 373 kB in 27s (13.6 kB/s)  
  
Selecting previously unselected package libcurl4:amd64.  
(Reading database ... 180045 files and directories currently installed.)  
Preparing to unpack .../libcurl4_7.58.0-2ubuntu3.8_amd64.deb ...  
Unpacking libcurl4:amd64 (7.58.0-2ubuntu3.8) ...  
Selecting previously unselected package curl.
```

Now, we can check the version of docker by typing the following command:

1. Docker -version

Output:

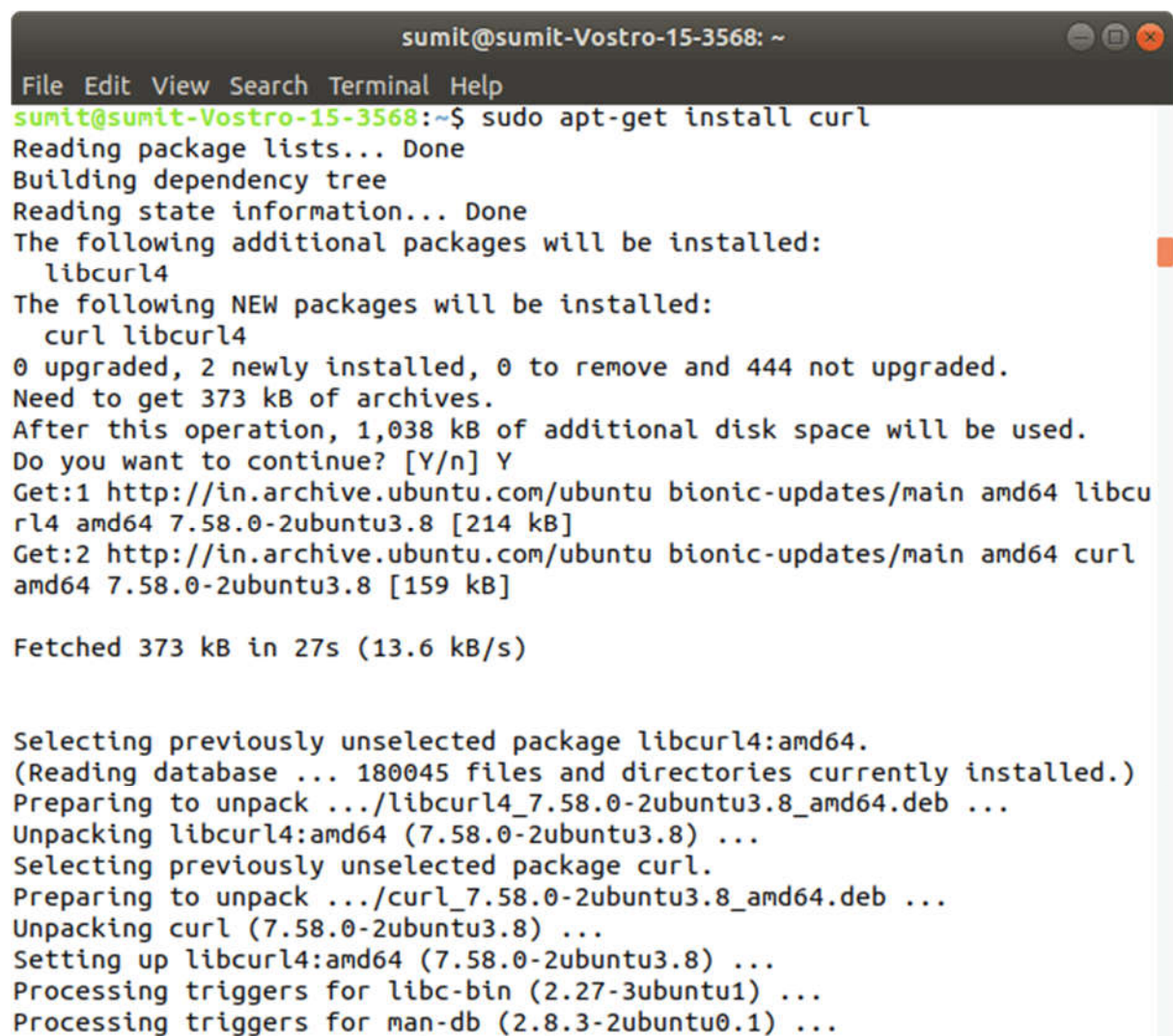
```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ docker --version  
Docker version 19.03.6, build 369ce74a3c  
sumit@sumit-Vostro-15-3568:~$
```

Step 3: After the successful execution of all the commands of the second step, we have to install the curl command. The curl is used to send the data using URL syntax.

Now, install the curl by using the following command. In the installation, we have to type Y.

1. `sudo apt-get install curl`

Output:

A terminal window titled 'sumit@sumit-Vostro-15-3568: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command 'sumit@sumit-Vostro-15-3568:~\$ sudo apt-get install curl' and its output. The output includes package list reading, dependency tree building, state information reading, and the installation of libcurl4 and curl. It shows the disk space requirements and the progress of downloading the packages. Finally, it shows the unpacking and setting up of the packages.

```
sumit@sumit-Vostro-15-3568: ~
File Edit View Search Terminal Help
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install curl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libcurl4
The following NEW packages will be installed:
  curl libcurl4
0 upgraded, 2 newly installed, 0 to remove and 444 not upgraded.
Need to get 373 kB of archives.
After this operation, 1,038 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcu
rl4 amd64 7.58.0-2ubuntu3.8 [214 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 curl
amd64 7.58.0-2ubuntu3.8 [159 kB]

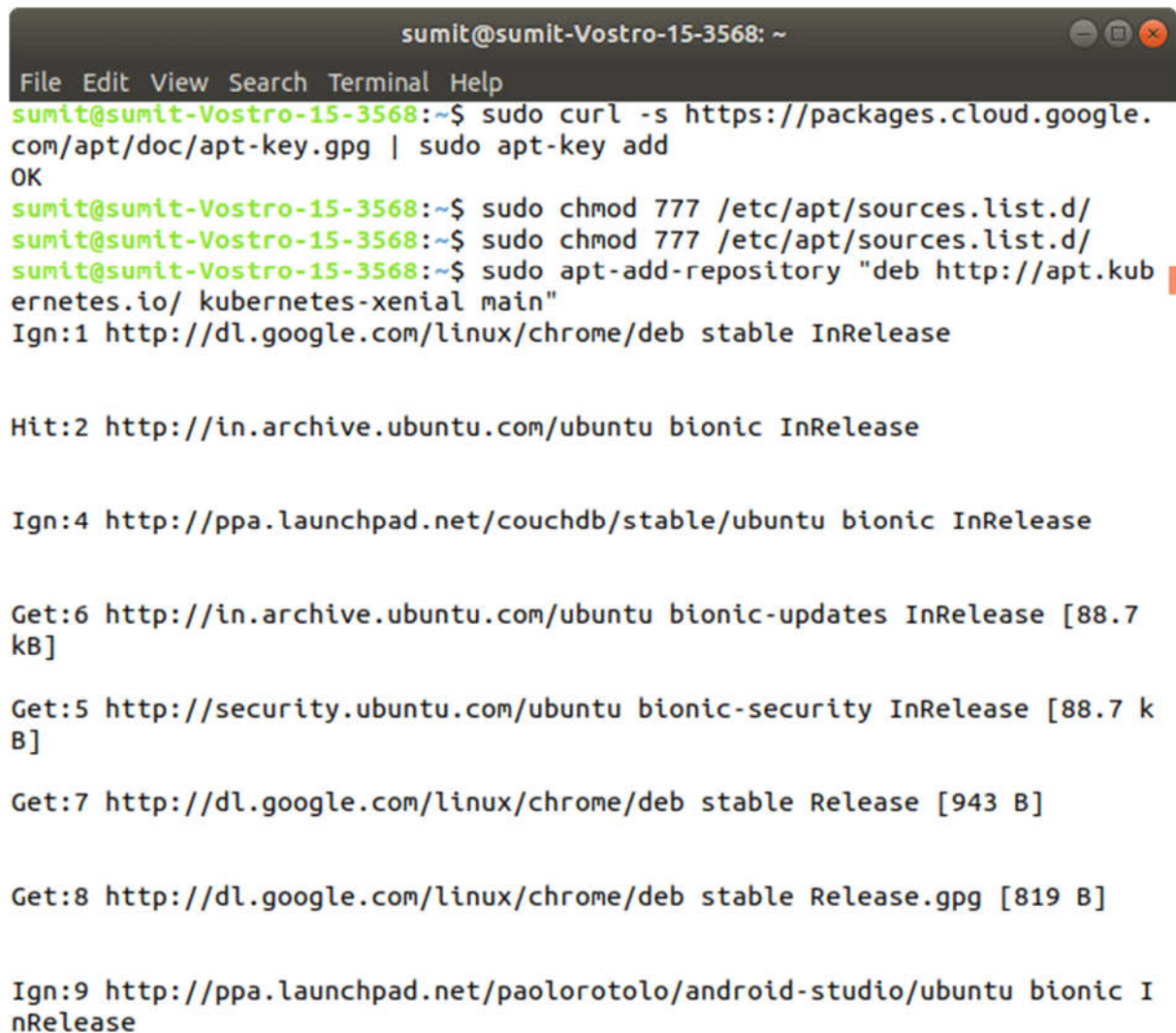
Fetched 373 kB in 27s (13.6 kB/s)

Selecting previously unselected package libcurl4:amd64.
(Reading database ... 180045 files and directories currently installed.)
Preparing to unpack .../libcurl4_7.58.0-2ubuntu3.8_amd64.deb ...
Unpacking libcurl4:amd64 (7.58.0-2ubuntu3.8) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_7.58.0-2ubuntu3.8_amd64.deb ...
Unpacking curl (7.58.0-2ubuntu3.8) ...
Setting up libcurl4:amd64 (7.58.0-2ubuntu3.8) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

Now, we have to download the add package key for Kubernetes by the following command:

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

Output:



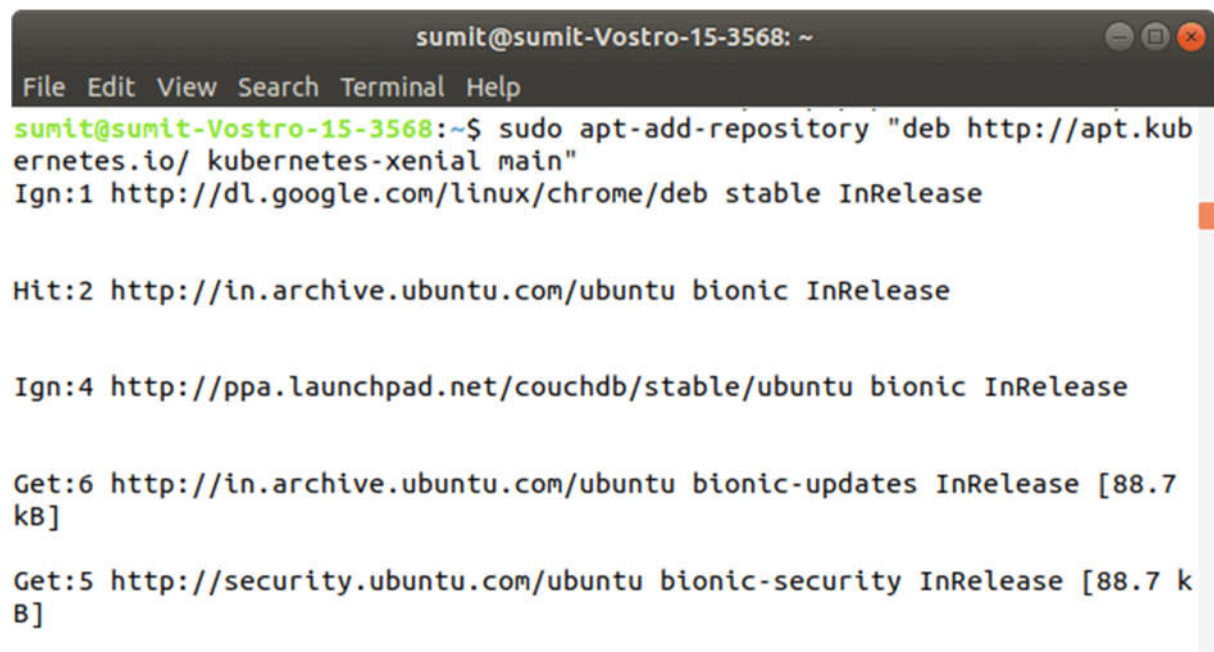
```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo curl -s https://packages.cloud.google.  
com/apt/doc/apt-key.gpg | sudo apt-key add  
OK  
sumit@sumit-Vostro-15-3568:~$ sudo chmod 777 /etc/apt/sources.list.d/  
sumit@sumit-Vostro-15-3568:~$ sudo chmod 777 /etc/apt/sources.list.d/  
sumit@sumit-Vostro-15-3568:~$ sudo apt-add-repository "deb http://apt.kub  
ernetes.io/ kubernetes-xenial main"  
Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease  
  
Hit:2 http://in.archive.ubuntu.com/ubuntu bionic InRelease  
  
Ign:4 http://ppa.launchpad.net/couchdb/stable/ubuntu bionic InRelease  
  
Get:6 http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7  
kB]  
Get:5 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 k  
B]  
Get:7 http://dl.google.com/linux/chrome/deb stable Release [943 B]  
  
Get:8 http://dl.google.com/linux/chrome/deb stable Release.gpg [819 B]  
  
Ign:9 http://ppa.launchpad.net/paolorotolo/android-studio/ubuntu bionic I  
nRelease
```

If you get an error from the above command, then it means your curl command is not successfully installed, so first install the curl command, and again run the above command.

Now, we have to add the Kubernetes repositories by the following command:

1. `sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"`

Output:

A terminal window titled 'sumit@sumit-Vostro-15-3568: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The command 'sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"' is entered. The output shows several repository updates: Ignoring 1 repository (http://dl.google.com/linux/chrome/deb stable InRelease), Hitting 2 repositories (http://in.archive.ubuntu.com/ubuntu bionic InRelease), Ignoring 4 repositories (http://ppa.launchpad.net/couchdb/stable/ubuntu bionic InRelease), Getting 6 repositories (http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]), and Getting 5 repositories (http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]).

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo apt-add-repository "deb http://apt.kub  
ernetes.io/ kubernetes-xenial main"  
Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease  
  
Hit:2 http://in.archive.ubuntu.com/ubuntu bionic InRelease  
  
Ign:4 http://ppa.launchpad.net/couchdb/stable/ubuntu bionic InRelease  
  
Get:6 http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7  
kB]  
Get:5 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 k  
B]
```

After the successful execution of the above command, we have to check any updates by executing the following command:

1. `sudo apt-get update`

Output:


```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo apt-get update  
Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease  
Ign:2 http://ppa.launchpad.net/couchdb/stable/ubuntu bionic InRelease  
  
Get:3 http://dl.google.com/linux/chrome/deb stable Release [943 B]  
  
Ign:4 http://ppa.launchpad.net/paolorotolo/android-studio/ubuntu bionic I  
nRelease  
  
Get:5 http://dl.google.com/linux/chrome/deb stable Release.gpg [819 B]  
  
Get:6 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 k  
B]  
  
Hit:8 http://in.archive.ubuntu.com/ubuntu bionic InRelease
```

Step 4: After the execution of the above commands in the above steps, we have to install the components of Kubernetes by executing the following command:

1. `sudo apt-get install -y kubelet kubeadm kubectl kubernetes-cni`

Output:

```
sumit@sumit-Vostro-15-3568: ~
File Edit View Search Terminal Help
sumit@sumit-Vostro-15-3568:~$ sudo apt-get install -y kubelet kubeadm kubectl
kubernetes-cni
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables ethtool socat
The following NEW packages will be installed:
  conntrack cri-tools ebtables ethtool kubeadm kubectl kubelet kubernetes-cni
  socat
0 upgraded, 9 newly installed, 0 to remove and 447 not upgraded.
Need to get 51.8 MB of archives.
After this operation, 273 MB of additional disk space will be used.
Get:4 http://in.archive.ubuntu.com/ubuntu bionic/main amd64 conntrack amd64 1
:1.4.4+snapshot20161117-6ubuntu2 [30.6 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 ebtables
amd64 2.0.10.4-3.5ubuntu2.18.04.3 [79.9 kB]
Get:1 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 cri-
```

Step 5: After the above installation is done, we have to initialize the kubeadm by executing the following command. The following command disables the swapping on other devices:

1. `sudo swapoff -a`

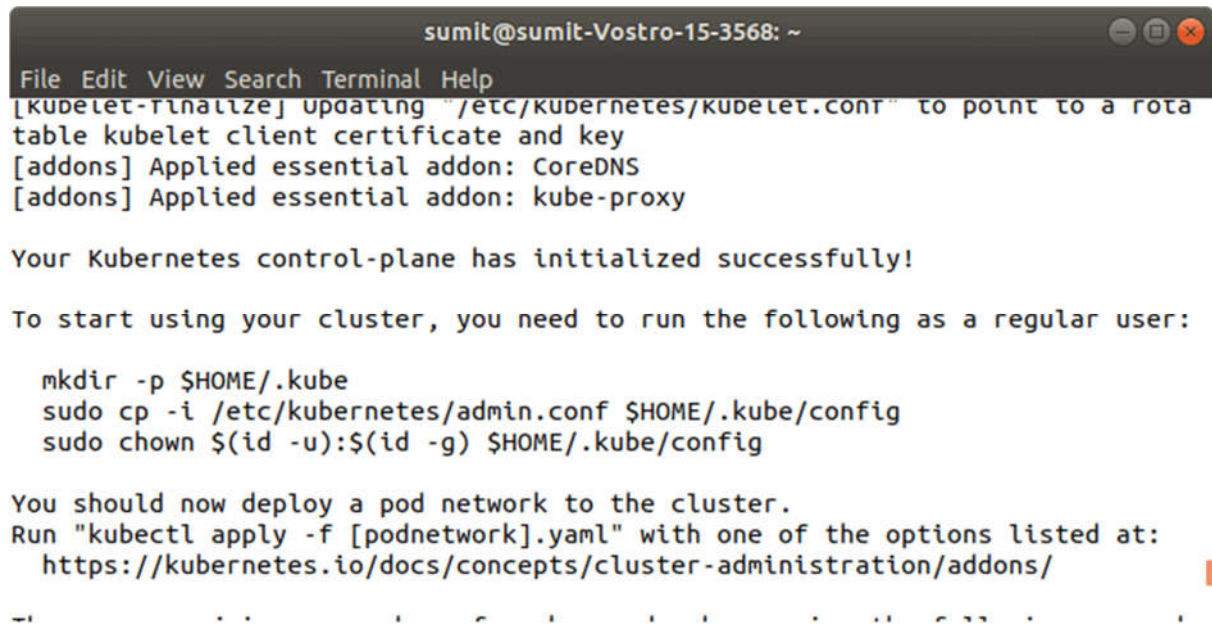
Output:

```
sumit@sumit-Vostro-15-3568: ~
File Edit View Search Terminal Help
sumit@sumit-Vostro-15-3568:~$ sudo swapoff -a
[sudo] password for sumit:
sumit@sumit-Vostro-15-3568:~$ █
```

Now, we have to initialize the kubeadm by executing the following command:

1. `sudo kubeadm init`

Output:

A terminal window titled 'sumit@sumit-Vostro-15-3568: ~' showing the output of the 'kubeadm init' command. The output includes messages about updating kubelet configuration, applying addons (CoreDNS and kube-proxy), and a success message. It also provides instructions for starting the cluster as a regular user, including creating a .kube directory and copying configuration files. Finally, it suggests deploying a pod network.

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
[kubelet-finalize] updating "/etc/kubernetes/kubelet.conf" to point to a rota  
table kubelet client certificate and key  
[addons] Applied essential addon: CoreDNS  
[addons] Applied essential addon: kube-proxy  
  
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/
```

Step 6: After the above command is successfully executed, we have to run the following commands, which are given in the initialization of kubeadm. These commands are shown in the above screenshot. The following commands are used to start a cluster:

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

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ mkdir -p $HOME/.kube  
sumit@sumit-Vostro-15-3568:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.  
kube/config  
[sudo] password for sumit:  
Skc@Sorry, try again.  
[sudo] password for sumit:  
sumit@sumit-Vostro-15-3568:~$ ^C  
sumit@sumit-Vostro-15-3568:~$ mkdir -p $HOME/.kube  
sumit@sumit-Vostro-15-3568:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.ku  
be/config  
cp: overwrite '/home/sumit/.kube/config'?  
sumit@sumit-Vostro-15-3568:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml  
podsecuritypolicy.policy/psp.flannel.unprivileged created  
clusterrole.rbac.authorization.k8s.io/flannel created  
clusterrolebinding.rbac.authorization.k8s.io/flannel created  
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml  
podsecuritypolicy.policy/psp.flannel.unprivileged created  
clusterrole.rbac.authorization.k8s.io/flannel created  
clusterrolebinding.rbac.authorization.k8s.io/flannel created  
serviceaccount/flannel created  
configmap/kube-flannel-cfg created  
daemonset.apps/kube-flannel-ds-amd64 created  
daemonset.apps/kube-flannel-ds-arm64 created  
daemonset.apps/kube-flannel-ds-arm created  
daemonset.apps/kube-flannel-ds-ppc64le created  
daemonset.apps/kube-flannel-ds-s390x created  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k8s-manifests/kube-flannel-rbac.  
yaml  
Unable to connect to the server: net/http: TLS handshake timeout
```

Output:

Step 7: In this step, we have to deploy the paths using the following command:

1. `sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml`

Output:

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml  
podsecuritypolicy.policy/psp.flannel.unprivileged created  
clusterrole.rbac.authorization.k8s.io/flannel created  
clusterrolebinding.rbac.authorization.k8s.io/flannel created  
serviceaccount/flannel created  
configmap/kube-flannel-cfg created  
daemonset.apps/kube-flannel-ds-amd64 created  
daemonset.apps/kube-flannel-ds-arm64 created  
daemonset.apps/kube-flannel-ds-arm created  
daemonset.apps/kube-flannel-ds-ppc64le created  
daemonset.apps/kube-flannel-ds-s390x created  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/k8s-manifests/kube-flannel-rbac.yml  
Unable to connect to the server: net/http: TLS handshake timeout  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl apply -f https://raw.githubusercontent.com
```

Step 8: After the execution of the above command, we have to run the following command to verify the installation:

1. `sudo kubectl get pods --all-namespaces`

Output:

```
sumit@sumit-Vostro-15-3568: ~  
File Edit View Search Terminal Help  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl get pods --all-namespaces  
NAMESPACE      NAME                                     READY   STATUS              RESTARTS   AGE  
kube-system     coredns-6955765f44-cs427               0/1     ContainerCreating   0          22m  
kube-system     coredns-6955765f44-s5c54               0/1     ContainerCreating   0          22m  
kube-system     etcd-sumit-vostro-15-3568             1/1     Running             0          22m  
kube-system     kube-apiserver-sumit-vostro-15-3568    1/1     Running             0          22m  
kube-system     kube-controller-manager-sumit-vostro-15-3568  1/1     Running             2          22m  
kube-system     kube-flannel-ds-amd64-5nt7f            0/1     CrashLoopBackOff    6          16m  
kube-system     kube-proxy-7vb2s                       1/1     Running             0          22m  
kube-system     kube-scheduler-sumit-vostro-15-3568    1/1     Running             2          22m  
sumit@sumit-Vostro-15-3568:~$  
sumit@sumit-Vostro-15-3568:~$  
sumit@sumit-Vostro-15-3568:~$ sudo kubectl get pods --all-namespaces
```

If the output is displayed as shown in the above screenshot. It means that the Kubernetes is successfully installed on our system.