|  |  |
| --- | --- |
| **DOCUMENT RULES:** | |
| **Task Number / Name:** | **KubeVirt** |
| **Task name & column name should be written:** | **Bold (CTRL+B)** |
| **Commands should be written in the after # sign:** | *Italic (CTRL+I) #hostname* |
| **Output photo should be cropped or compressed:**  **Photo could be more than one:**  **If you need extra lines, add the line next after it:** | ***Description photo should be with title bar (CTRL + I + B)*** |
| **All other text should be written:** | Standard |
| **Font name and text size:** | Calibri and 9 |
| **Group name:** | Dev\_ops\_ |
| **Student name and surname:** | Murad Abbaszade |
| **E-mail:** | [muradabbaszade6@gmail.com](mailto:muradabbaszade6@gmail.com) |
| **WhatsApp number:** | **+994703664205** |

Introduction to KubeVirt

## Deploy KubeVirt

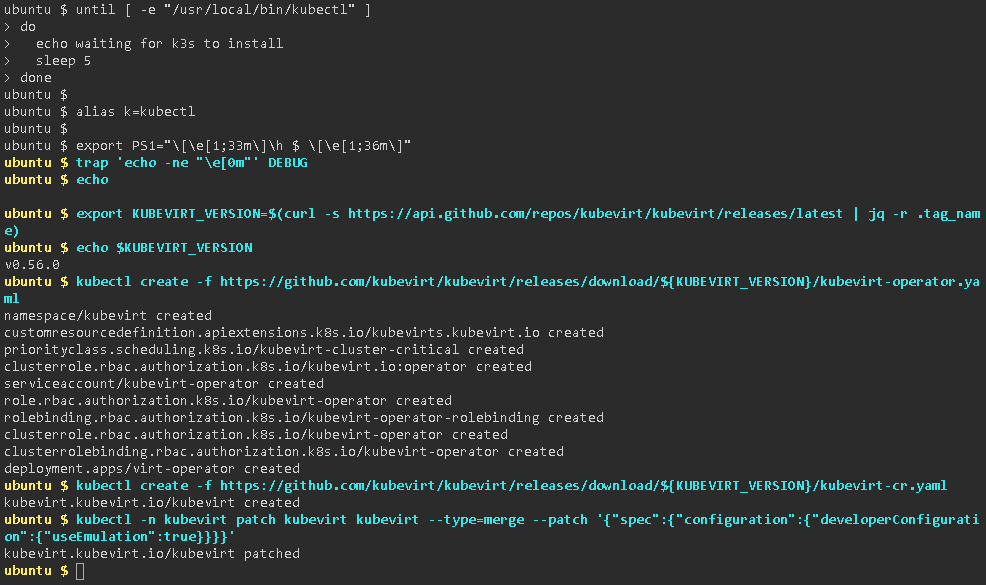
export KUBEVIRT\_VERSION=$(curl -s https://api.github.com/repos/kubevirt/kubevirt/releases/latest | jq -r .tag\_name)

echo $KUBEVIRT\_VERSION

kubectl create -f [https://github.com/kubevirt/kubevirt/releases/download/${KUBEVIRT\_VERSION}/kubevirt-operator.yaml](https://github.com/kubevirt/kubevirt/releases/download/$%7bKUBEVIRT_VERSION%7d/kubevirt-operator.yaml)

kubectl create -f [https://github.com/kubevirt/kubevirt/releases/download/${KUBEVIRT\_VERSION}/kubevirt-cr.yaml](https://github.com/kubevirt/kubevirt/releases/download/$%7bKUBEVIRT_VERSION%7d/kubevirt-cr.yaml)

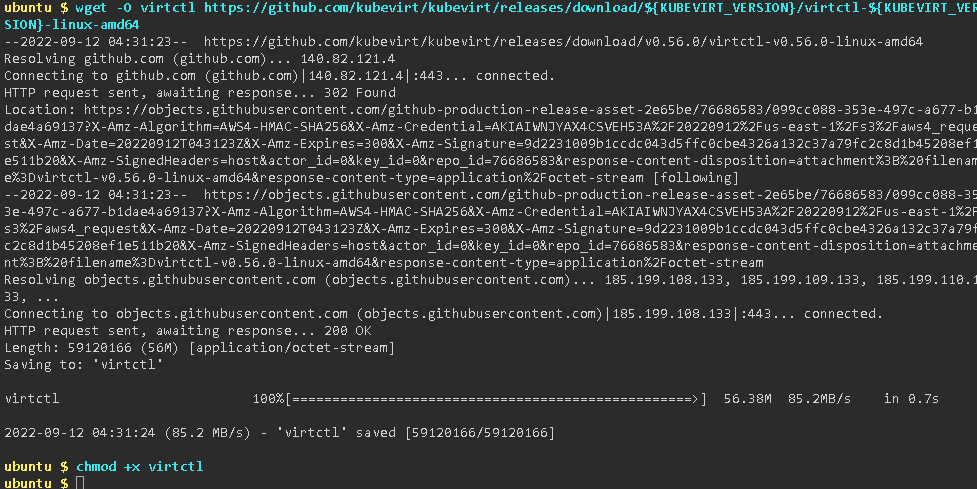
kubectl -n kubevirt patch kubevirt kubevirt --type=merge --patch '{"spec":{"configuration":{"developerConfiguration":{"useEmulation":true}}}}'



# Install Virtctl

wget -O virtctl https://github.com/kubevirt/kubevirt/releases/download/${KUBEVIRT\_VERSION}/virtctl-${KUBEVIRT\_VERSION}-linux-amd64

chmod +x virtctl



# Wait for KubeVirt deployment to finalize

kubectl get pods -n kubevirt

kubectl -n kubevirt get kubevirt



# Deploy a Virtual Machine

kubectl apply -f <https://kubevirt.io/labs/manifests/vm.yaml>

kubectl get vms

kubectl get vms -o yaml testvm | grep -E 'running:.\*|$'

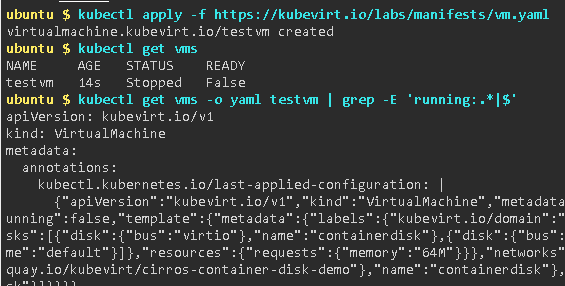
To start a VM, use virtctl with the start verb:

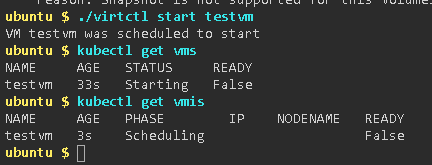
./virtctl start testvm

Again, check the VM status:

kubectl get vms

kubectl get vmis





# Access a VM (serial console & vnc)

**NOTE:** ^] means: press the "CTRL" and "]" keys to escape the console.

./virtctl console testvm



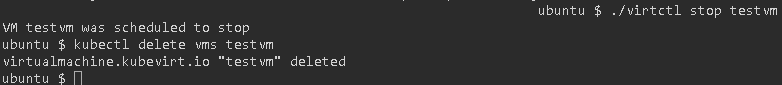
# Shutdown and cleanup

As with starting, stopping a VM also may be accomplished with the virtctl command:

./virtctl stop testvm

Finally, the VM can be deleted as any other Kubernetes resource using kubectl:

kubectl delete vms testvm



# Experiment with CDI

## Install CDI

# Check Default Storage Class

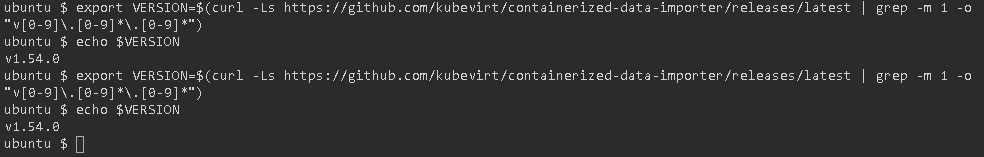
# kubectl get storageclass



# Install the Containerized Data Importer

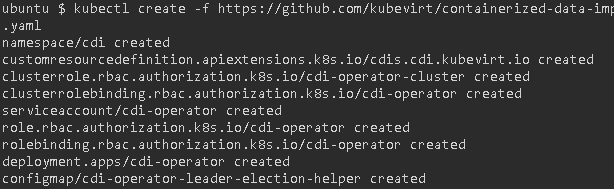
export VERSION=$(curl -Ls https://github.com/kubevirt/containerized-data-importer/releases/latest | grep -m 1 -o "v[0-9]\.[0-9]\*\.[0-9]\*")

echo $VERSION



kubectl create -f https://github.com/kubevirt/containerized-data-importer/releases/download/$VERSION/cdi-operator.yaml

kubectl -n cdi scale deployment/cdi-operator --replicas=1





kubectl create -f <https://github.com/kubevirt/containerized-data-importer/releases/download/$VERSION/cdi-cr.yaml>



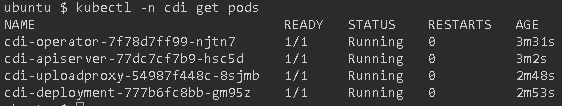
kubectl get cdi -n cdi



kubectl wait -n cdi --for=jsonpath='{.status.phase}'=Deployed cdi/cdi



kubectl -n cdi get pods



## Use CDI to import and start a VM

# Use CDI to upload a VM image

cat <<EOF > pvc\_cirros.yml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: "cirros"

labels:

app: containerized-data-importer

annotations:

cdi.kubevirt.io/storage.import.endpoint: "http://download.cirros-cloud.net/0.5.2/cirros-0.5.2-x86\_64-disk.img"

kubevirt.io/provisionOnNode: node01

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 120Mi

EOF

kubectl create -f pvc\_cirros.yml

# 

# kubectl get pod

# 

# kubectl logs importer-cirros –f

# 

cat <<EOF > vm1.yml

apiVersion: kubevirt.io/v1

kind: VirtualMachine

metadata:

labels:

kubevirt.io/os: linux

name: vm1

spec:

running: true

template:

metadata:

creationTimestamp: null

labels:

kubevirt.io/domain: vm1

spec:

domain:

cpu:

cores: 1

devices:

disks:

- disk:

bus: virtio

name: disk0

- cdrom:

bus: sata

readonly: true

name: cloudinitdisk

resources:

requests:

memory: 128M

volumes:

- name: disk0

persistentVolumeClaim:

claimName: cirros

- cloudInitNoCloud:

userData: |

#cloud-config

user: cirros

password: gocubsgo

hostname: vm1

ssh\_pwauth: True

disable\_root: false

name: cloudinitdisk

EOF

kubectl create -f vm1.yml

# 

# 

# kubectl get pod -o wide

# 

# kubectl get vmi

# 

# IP=$(kubectl get vmi vm1 -o jsonpath='{.status.interfaces[0].ipAddress}')

# ssh cirros@${IP}

# 

# ssh-copy-id -i ~/.ssh/id\_rsa.pub cirros@${IP}

# 

# ssh cirros@${IP} hostname

# 

# virtctl stop vm1

# 

# kubectl get vmi

# 

# virtctl start vm1

# 

# IP=$(kubectl get vmi vm1 -o jsonpath='{.status.interfaces[0].ipAddress}')

# ssh cirros@${IP} hostname

# 