云原生云主机服务（KubeVirt）使用手册 Lite

准备工作

安装 virtctl

```
wget <https://github.com/kubevirt/kubevirt/releases/download/v0.47.1/virtctl-v0.47.1-linux-amd64>
chmod a+x virtctl-v0.47.1-linux-amd64
cp virtctl-v0.47.1-linux-amd64 /usr/local/bin/virtctl
```

下载镜像

CentOS：

CentOS Linux and Stream cloud images download

公司自制镜像：

> 注意：自制的CentOS镜像kubevirt不可用

X86-标准镜像维护列表

Microsoft官方Windows ISO镜像：

Microsoft Evaluation Center

安装服务

云产品：云原生云主机安装

过程略

使用

镜像管理

上传镜像

云原生云主机服务通过CDI模块实现上传镜像的能力

执行以下命令上传镜像：

```
virtctl image-upload pvc <PVC名称> --size=10Gi --image-path=<本地镜像文件路径> --storage-class=<存储类> --wait-secs=240 --uploadproxy-url=<https://cdi-uploa
```

CDI会自动创建一个指定PVC名称的PVC来保存镜像。

删除镜像

删除对应PVC即可：

```
kubectl delete pvc <PVC名称>
```

~~预设管理（将被弃用）~~

> 预设仅支持定义domain属性

创建预设

创建 `5.vmipreset-small.yaml` 如下：

```yaml
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachineInstancePreset
metadata:
  name: c2-g4
spec:
  selector:
    matchLabels:
      kubevirt.io/size: 2C-4G
  domain:
    cpu:
      cores: 2
    resources:
      requests:
        memory: 4G
    devices: {}
```

查看预设

```
kubectl get virtualmachineinstancepresets
```

修改预设

```
kubectl edit virtualmachineinstancepresets c2-g4
```

删除预设

```
kubectl delete virtualmachineinstancepresets c2-g4
```

云主机管理

创建云主机

云原生云主机的属性均通过CR VirtualMachine定义。

配置项

设置CPU：

```yaml
spec:
  template:
    spec:
      domain:
        cpu:
          cores: 2
```

设置内存：

```yaml
spec:
  template:
    spec:
      domain:
        resources:
          requests:
            memory: 4G
```

设置存储：

- 卷设置：
  - PVC

    ```
    spec:
      template:
        spec:
          volumes:
          - name: sys-disk
            persistentVolumeClaim:
              claimName: centos-7.9
    ```

- 盘设置：
  - disk

    ```
    spec:
      template:
        spec:
          domain:
            devices:
              disks:
                # This makes it a disk
              - disk:
                  # This makes it exposed as /dev/vda, being the only and thus first
                  # disk attached to the VM
                  bus: virtio
                name: sys-disk
    ```

  - cdrom

    ```
    spec:
      template:
        spec:
          domain:
            devices:
              disks:
              - name: cdromiso
                # This makes it a cdrom
                cdrom:
                  # This makes the cdrom writeable
                  readOnly: false
                  # This makes the cdrom be exposed as SATA device
                  bus: sata
    ```

设置启动顺序：

```
spec:
  template:
    spec:
      domain:
        devices:
          disks:
          - bootOrder: 1
```

```
              cdrom:
                bus: sata
              name: cdromiso
            - bootOrder: 2
              disk:
                bus: virtio
              name: harddrive
```

设置网络：

- 接口设置：
    - slirp

      略
    - macvtap

      略
- 网络设置：
    -
- 固定IP
    - pod默认网络：

      当前pod默认网络使用为flannel，不支持固定IP
    - 集群第二网络：
        - kube-ovn

          kube-ovn的新特性已经支持云主机的生命周期中自动分配并固定IP与Mac地址（详见：EAS-103081 ）。若需要手动指定IP与Mac地址，可通过以下方式：

          ```
          spec:
            template:
              metadata:
                annotations:
                  ovn.kubernetes.io/ip_address: <ip地址>
                  ovn.kubernetes.io/mac_address: <mac地址>
          ```
        - sriov-cni/ib-sriov-cni

          不支持固定IP

使用预设：

```
spec:
  template:
    metadata:
      labels:
        kubevirt.io/size: 2C-4G
```

秘钥注入：

首先创建SSH公钥Secret：

```
apiVersion: v1
data:
  key1: <ssh_public_key>
kind: Secret
metadata:
  name: public-key
type: Opaque
```

通过access credentials接口使用configDrive传递方式注入：

```
spec:
 template:
  spec:
   domain:
    devices:
     disks:
     - disk:
       bus: virtio
      name: cloudinitdisk
   accessCredentials:
   - sshPublicKey:
     propagationMethod:
      configDrive: {}
     source:
      secret:
       secretName: public-key
   volumes:
   - cloudInitConfigDrive:
     userData: |
      #cloud-config
      disable_root: false
    name: cloudinitdisk
```

密码注入：

用户，密码，以及其他初始化命令可通过`cloudInitNoCloud`卷的方式注入：

> 该方式也可注入秘钥，但上文秘钥注入方式相较于此的好处是与cloud-init用户数据（user data）隔离

```
spec:
 template:
  spec:
   domain:
    devices:
     disks:
     - disk:
       bus: virtio
      name: cloudinitdisk
   volumes:
   - cloudInitNoCloud:
     userData: |
      #cloud-config
      users:
       - name: escore
        gecos: ES Core User
        sudo: ALL=(ALL) NOPASSWD:ALL
        passwd: $6$xovHFad3iYjLc80I$jCog26PDU3r4BKBxFS3bejiglTyTyvldqK7hiExVP8rgbIbJh3CDXfs2GkmH038g4EW2KPsICnV8P6rHe1kcA/
        shell: /bin/bash
        home: /home/escore
        lock_passwd: false
        ssh_pwauth: true
    name: cloudinitdisk
```

创建 Linux 云主机

示例一：创建从PVC启动，使用默认Pod网络的云主机

> 该PVC通过CDI上传镜像创建，已含有镜像文件

`1.vm-centos-boot-pvc.yaml` ：

```yaml
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  name: centos-boot-pvc
spec:
  running: true
  template:
    metadata:
      labels:
        kubevirt.io/domain: centos-boot-pvc
    spec:
      domain:
        cpu:
          cores: 2
        devices:
          disks:
          - disk:
              bus: virtio
            name: boot-disk
          interfaces:
          - masquerade: {}
            name: default
        machine:
          type: q35
        resources:
          requests:
            memory: 4G
      networks:
      - name: default
        pod: {}
      volumes:
      - name: boot-disk
        persistentVolumeClaim:
          claimName: centos-7.9
```

示例二：创建从DV启动，使用集群第二网络的云主机

`2.vm-centos-net-ovn-boot-dv.yaml` ：

```yaml
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  name: centos-net-ovn-boot-dv
spec:
  running: true
  template:
    metadata:
      labels:
        kubevirt.io/domain: centos-net-ovn-boot-dv
    spec:
      domain:
        cpu:
          cores: 2
        devices:
          disks:
```

```
        - disk:
            bus: virtio
          name: boot-disk
        - disk:
            bus: virtio
          name: cloudinitdisk
        interfaces:
        - bridge: {}
          name: default
      machine:
        type: q35
      resources:
        requests:
          memory: 4G
    networks:
    - name: default
      multus:
        default: true
        networkName: eks-managed/kube-ovn
    volumes:
    - name: boot-disk
      dataVolume:
        name: centos-dv
    - cloudInitNoCloud:
        userData: |
          #cloud-config
          users:
            - name: escore
              gecos: ES Core User
              sudo: ALL=(ALL) NOPASSWD:ALL
              passwd: $6$xovHFad3iYjLc80I$jCog26PDU3r4BKBxFS3bejiglTyTyvldqK7hiExVP8rgbIbJh3CDXfs2GkmH038g4EW2KPsICnV8P6rHe1kcA/
              shell: /bin/bash
              home: /home/escore
              lock_passwd: false
              ssh_pwauth: true
      name: cloudinitdisk
dataVolumeTemplates:
- metadata:
    name: centos-dv
  spec:
    pvc:
      accessModes:
      - ReadWriteOnce
      resources:
        requests:
          storage: 30Gi
      storageClassName: capacity
    source:
      pvc:
        namespace: default
        name: centos-7.9
```

该云主机配置为集群第二网络：kube-ovn，通过DV模板定义的DV启动，此外还配置了 `cloudInitNoCloud` 卷用于注入自定义用户与密码。

示例三：创建使用预设的云主机

`5.vm-centos-boot-dv-with-preset.yaml` ：

```
apiVersion: kubevirt.io/v1alpha3
```

```yaml
kind: VirtualMachine
metadata:
  name: centos-boot-dv2
spec:
  running: true
  template:
    metadata:
      labels:
        kubevirt.io/domain: centos-boot-dv2
        kubevirt.io/size: 2C-4G
    spec:
      domain:
        devices:
          disks:
          - disk:
              bus: virtio
            name: boot-disk
          - disk:
              bus: virtio
            name: cloudinitdisk
          interfaces:
          - masquerade: {}
            name: default
        machine:
          type: q35
      networks:
      - name: default
        pod: {}
      volumes:
      - name: boot-disk
        dataVolume:
          name: centos-dv2
      - cloudInitNoCloud:
          userData: |
            #cloud-config
            users:
              - name: escore
                gecos: ES Core User
                sudo: ALL=(ALL) NOPASSWD:ALL
                passwd: $6$xovHFad3iYjLc80I$jCog26PDU3r4BKBxFS3bejiglTyTyvldqK7hiExVP8rgbIbJh3CDXfs2GkmH038g4EW2KPsICnV8P6rHe1kcA/
                shell: /bin/bash
                home: /home/escore
                lock_passwd: false
                ssh_pwauth: true
        name: cloudinitdisk
  dataVolumeTemplates:
  - metadata:
      name: centos-dv2
    spec:
      pvc:
        accessModes:
        - ReadWriteOnce
        resources:
          requests:
            storage: 30Gi
        storageClassName: capacity
      source:
        pvc:
          namespace: default
          name: centos-7.9
```

该云主机通过 labels 指定**预设**配置，注入了CPU与内存的规格。

示例四：创建使用sriov网络的云主机（手动）

> 该方式需要手动规划网络

通过cloud-init NoCloud方式注入网络配置：

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  name: centos-sriov
spec:
  dataVolumeTemplates:
    - metadata:
        creationTimestamp: null
        name: centos-sriov-dv
      spec:
        pvc:
          accessModes:
            - ReadWriteMany
          resources:
            requests:
              storage: 30Gi
          storageClassName: nfs-client
        source:
          pvc:
            name: centos-7.9
            namespace: nsa
  running: true
  template:
    metadata:
      labels:
        kubevirt.io/domain: centos-sriov
    spec:
      domain:
        cpu:
          cores: 2
        devices:
          disks:
            - disk:
                bus: virtio
              name: boot-disk
            - disk:
                bus: virtio
              name: cloudinitdisk
          interfaces:
            - bridge: {}
              name: ovn-net
            - sriov: {}
              macAddress: de:ad:00:00:be:aa
              name: sriov-net
        machine:
          type: q35
        resources:
          requests:
            memory: 4G
      networks:
        - multus:
            default: true
            networkName: eks-managed/kube-ovn
          name: ovn-net
        - multus:
            networkName: eks-managed/sriov-network
          name: sriov-net
      volumes:
        - dataVolume:
            name: centos-sriov-dv
          name: boot-disk
        - cloudInitNoCloud:
            userData: |
              #cloud-config
              disable_root: false
              ssh_pwauth: true
              users:
                - name: escore
                  gecos: ES Core User
                  sudo: ALL=(ALL) NOPASSWD:ALL
                  passwd: $6$xovHFad3iYjLc80I$jCog26PDU3r4BKBxFS3bejiglTyTyvldqK7hiExVP8rgbIbJh3CDXfs2GkmH038g4EW2KPsICnV8P6rHe1kcA/
                  shell: /bin/bash
                  home: /home/escore
                  lock_passwd: false
                  ssh_pwauth: true
              runcmd:
                - dhclient eth1
            networkData: |
              version: 2
              ethernets:
                sriov-net:
                  set-name: eth0
                  match:
                    macaddress: 'de:ad:00:00:be:aa'
                  addresses:
                    - 10.37.1.64/24
          name: cloudinitdisk
```

NoCloud网络配置格式规范详见：Networking Config Version 1，[Networking Config Version 2](https://cloudinit.readthedocs.io/en/latest/topics/network-config-format-v2.html)

通过cloud-init ConfigDrive方式注入网络配置：

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  name: centos-sriov
spec:
  dataVolumeTemplates:
    - metadata:
        creationTimestamp: null
        name: centos-sriov-dv
      spec:
        pvc:
          accessModes:
            - ReadWriteMany
          resources:
```

```
          requests:
            storage: 30Gi
        storageClassName: nfs-client
      source:
        pvc:
          name: centos-7.9
          namespace: nsa
running: true
template:
  metadata:
    labels:
      kubevirt.io/domain: centos-sriov
  spec:
    domain:
      cpu:
        cores: 2
      devices:
        disks:
          - disk:
              bus: virtio
            name: boot-disk
          - disk:
              bus: virtio
            name: cloudinitdisk
        interfaces:
          - bridge: {}
            name: ovn-net
          - sriov: {}
            macAddress: de:ad:00:00:be:aa
            name: sriov-net
      machine:
        type: q35
      resources:
        requests:
          memory: 4G
    networks:
      - multus:
          default: true
          networkName: eks-managed/kube-ovn
        name: ovn-net
      - multus:
          networkName: eks-managed/sriov-network
        name: sriov-net
    volumes:
      - dataVolume:
          name: centos-sriov-dv
        name: boot-disk
      - cloudInitConfigDrive:
          userData: |
            #cloud-config
            disable_root: false
            ssh_pwauth: true
            users:
              - name: escore
                gecos: ES Core User
                sudo: ALL=(ALL) NOPASSWD:ALL
                passwd: $6$xovHFad3iYjLc80I$jCog26PDU3r4BKBxFS3bejiglTyTyvldqK7hiExVP8rgbIbJh3CDXfs2GkmH038g4EW2KPsICnV8P6rHe1kcA/
                shell: /bin/bash
                home: /home/escore
                lock_passwd: false
                ssh_pwauth: true
            runcmd:
              - dhclient eth1
          networkData: |
            {"networks": [{"network_id": "84c285d6-286c-4701-870f-96f99589c9b3", "type": "ipv4", "netmask": "255.255.255.0", "link": "interface0", "routes": [{"netma
        name: cloudinitdisk
```

ConfigDrive网络配置格式规范详见：OpenStack Metadata Service Network: Sample API

创建 Windows 云主机

示例一： iso 启动

首先需要创建一个卷作为云主机的数据盘，PVC 示例 `1.pvc-winhd.yaml` 如下：

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: winhd
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 150Gi
  storageClassName: capacity
```

云主机示例 `1.vm-win2k12-boot-iso.yaml` 如下：

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  name: win2k12-boot-iso
spec:
  running: true
  template:
    metadata:
      labels:
        kubevirt.io/domain: win2k12-boot-iso
    spec:
      domain:
        cpu:
          cores: 4
        devices:
          disks:
          - bootOrder: 1
            cdrom:
              bus: sata
            name: cdromiso
          - disk:
              bus: virtio
            name: harddrive
          - cdrom:
              bus: sata
            name: virtiocontainerdisk
          interfaces:
          - masquerade: {}
            name: default
        machine:
          type: q35
        resources:
          requests:
            memory: 8G
      networks:
      - name: default
        pod: {}
      volumes:
      - name: cdromiso
        persistentVolumeClaim:
          claimName: iso-win-2k12-zh
      - name: harddrive
        persistentVolumeClaim:
          claimName: winhd
      - containerDisk:
          image: hub.easystack.io/production/virtio-container-disk
        name: virtiocontainerdisk
```

容器盘`hub.easystack.io/production/virtio-container-disk`包含了 Windows 云主机所需驱动。

安装过程参考：KubeVirt: installing Microsoft Windows from an ISO

示例二： raw 启动

云主机示例 `2.vm-win2k19-boot-dv.yaml` 如下：

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  name: win2k19-boot-dv
spec:
  running: true
  template:
    metadata:
      labels:
        kubevirt.io/domain: win2k19-boot-dv
    spec:
      domain:
        cpu:
          cores: 4
        devices:
          disks:
          - disk:
              bus: virtio
            name: boot-disk
          interfaces:
          - masquerade: {}
            name: default
        machine:
          type: q35
        resources:
          requests:
            memory: 8G
      networks:
      - name: default
        pod: {}
      volumes:
      - name: boot-disk
        dataVolume:
          name: win2k19-dv
  dataVolumeTemplates:
  - metadata:
      name: win2k19-dv
    spec:
      pvc:
        accessModes:
        - ReadWriteOnce
        resources:
          requests:
            storage: 50Gi
        storageClassName: capacity
      source:
        pvc:
          namespace: default
          name: win-2k19-en
```

批量创建云主机

> VirtualMachineInstanceReplicaSet 允许你在 KubeVirt 中批量创建和管理虚拟机实例。它具有以下重要字段和功能：
>
> spec.selector：用于选择要管理的虚拟机实例的标签选择器。只有与该标签选择器匹配的虚拟机实例才会被 ReplicaSet 管理。
> spec.template：定义了创建虚拟机实例的模板，包括虚拟机的规格、存储、网络等配置信息。
> spec.replicas：指定要创建的虚拟机实例的数量。
> status.replicas：表示当前实际运行的虚拟机实例的数量。
> 当你创建一个 VirtualMachineInstanceReplicaSet 时，KubeVirt 会自动根据 ReplicaSet 的定义，创建指定数量的虚拟机实例。如果有虚拟机实例的数量不符合定义，KubeVirt 会自动进行扩缩容操作，以确保虚拟机实例的数量与定义的数量一致。
>
> 你可以使用 kubectl 或其他 Kubernetes 客户端工具，以 YAML 或 JSON 格式定义 VirtualMachineInstanceReplicaSet 对象，并将其应用到 Kubernetes 集群中。

创建 VirtualMachineInstanceReplicaSet

```
apiVersion: kubevirt.io/v1
kind: VirtualMachineInstanceReplicaSet
metadata:
  name: ubuntureplicaset
spec:
  replicas: 3   #代表创建3副本虚拟机
  selector:
    matchLabels:
      kubevirt.io/domain: ubuntu-ovn
  template:
    metadata:
      annotations:
        ovn.kubernetes.io/allow_live_migration: 'true'
      labels:
        kubevirt.io/domain: ubuntu-ovn
    spec:
      accessCredentials:
        - sshPublicKey:
            propagationMethod:
              configDrive: {}
            source:
              secret:
                secretName: env35
      domain:
        cpu:
```

```
      cores: 2
     devices:
      disks:
       - disk:
         bus: virtio
        name: boot-disk
       - disk:
         bus: virtio
        name: cloudinitdisk
      interfaces:
       - name: default
        bridge: {}
     machine:
      type: q35
     resources:
      requests:
       memory: 4G
    networks:
     - name: default
      multus:
       default: true
       networkName: eks-managed/kube-ovn
    volumes:
     - name: boot-disk
      containerDisk:
       image: kubevirt/fedora-cloud-container-disk-demo:latest
       imagePullPolicy: IfNotPresent
     - cloudInitConfigDrive:
       userData: |
        #cloud-config
        disable_root: false
        ssh_pwauth: true
        users:
         - name: escore
          gecos: ES Core User
          sudo: ALL=(ALL) NOPASSWD:ALL
          passwd: $6$xovHFad3iYjLc80I$jCog26PDU3r4BKBxFS3bejiglTyTyvldqK7hiExVP8rgbIbJh3CDXfs2GkmH038g4EW2KPsICnV8P6rHe1kcA/
          shell: /bin/bash
          home: /home/escore
          lock_passwd: false
          ssh_pwauth: true
      name: cloudinitdisk
```

通过容器镜像的方式创建云主机

用户可以将VirtualMachineInstance磁盘注入到容器映像中，这种方式可以被KubeVirt运行时使用。磁盘必须放置在容器内的/disk目录下。支持raw和qcow2格式。为了减小容器映像的大小，建议使用qcow2。containerdisks可以并且应该基于scratch。

制作容器镜像

将本地VirtualMachineInstance磁盘注入到容器映像中，制作成启动镜像。
Dockerfile示例如下

```
FROM scratch
ADD --chown=107:107 ubuntu.qcow2 /disk/
```

生成镜像：

```
docker build -t vmidisks/ubuntu:latest .
```

空硬盘批量创建

> emptyDisk在批量创建云主机时挂载空硬盘

例：创建并挂在一个2Gib的云硬盘

```yaml
spec:
  terminationGracePeriodSeconds: 5
  domain:
    resources:
      requests:
        memory: 64M
    devices:
      disks:
      - name: containerdisk
        disk:
          bus: virtio
      - name: emptydisk
        disk:
          bus: virtio
  volumes:
    - name: containerdisk
      containerDisk:
        image: kubevirt/cirros-registry-disk-demo:latest
    - name: emptydisk
      emptyDisk:
        capacity: 2Gi
```

挂载configmap

> KubeVirt 将会将 ConfigMap 中的内容挂载到虚拟机实例的指定路径中。您可以根据自己的需求，调整文件的路径和名称。

```yaml
spec:
  domain:
    devices:
      disks:
      - disk:
          bus: virtio
        name: containerdisk
      - disk:
          bus: virtio
        name: cloudinitdisk
      - disk:
        name: app-config-disk
        # set serial
        serial: CVLY623300HK240D
    machine:
      type: ""
    resources:
      requests:
        memory: 1024M
  terminationGracePeriodSeconds: 0
  volumes:
  - name: containerdisk
    containerDisk:
      image: kubevirt/fedora-cloud-container-disk-demo:latest
  - cloudInitNoCloud:
      userData: |-
```

```
          #cloud-config
          password: fedora
          chpasswd: { expire: False }
          bootcmd:
            # mount the ConfigMap
            - "sudo mkdir /mnt/app-config"
            - "sudo mount /dev/$(lsblk --nodeps -no name,serial | grep CVLY623300HK240D | cut -f1 -d' ') /mnt/app-config"
      name: cloudinitdisk
    - configMap:
        name: app-config
      name: app-config-disk
```

查看云主机实例

```
kubectl get virtualmachineinstances
```

云主机操作

电源管理

云主机启动

```
virtctl start centos-net-ovn-boot-dv
```

云主机关机

```
virtctl stop centos-net-ovn-boot-dv
```

云主机暂停

```
virtctl pause vm centos-net-ovn-boot-dv
```

云主机恢复

```
virtctl unpause vm centos-net-ovn-boot-dv
```

云主机重启

```
virtctl restart centos-net-ovn-boot-dv
```

配置管理

调整规格

```
kubectl edit virtualmachines centos-net-ovn-boot-dv
# 编辑 cpu, memory 字段
virtctl restart centos-net-ovn-boot-dv
```

通过预设调整规格

新建预设：`5.vmipreset-medium.yaml`：

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachineInstancePreset
metadata:
  name: c4-g8
spec:
  selector:
    matchLabels:
      kubevirt.io/size: 4C-8G
  domain:
```

```yaml
    cpu:
      cores: 4
    resources:
      requests:
        memory: 8G
    devices: {}
```

编辑 `spec.template.metadata.labels` 字段：

```
kubectl edit virtualmachines centos-boot-dv2
```

```yaml
spec:
  template:
    metadata:
      creationTimestamp: null
      labels:
        kubevirt.io/domain: centos-boot-dv2
        # kubevirt.io/size: 2C-4G
        kubevirt.io/size: 4C-8G
```

重启云主机

```
virtctl restart centos-boot-dv2
```

调整启动顺序

```
kubectl edit virtualmachines centos-boot-iso
# 编辑 bootOrder 字段
virtctl restart centos-boot-iso
```

存储管理

热插拔

先创建一个空DV：`3.dv-hotplug.yaml`

```yaml
apiVersion: cdi.kubevirt.io/v1beta1
kind: DataVolume
metadata:
  name: disk-hotplug
spec:
  source:
    blank: {}
  pvc:
    accessModes:
      - ReadWriteOnce
    resources:
      requests:
        storage: 5Gi
    storageClassName: capacity
```

- 卸载云硬盘

  ```
  virtctl removevolume centos-boot-dv --volume-name=disk-hotplug
  ```

维护

热迁移

```
virtctl migrate centos-boot-dv
```

冷迁移

云主机关机后，编译VirtualMachine添加以下字段指定目的节点：

```
spec:
  template:
    spec:
      nodeSelector:
        kubernetes.io/hostname: <hostname>
```

远程连接

> 注意：集群中需要打通相关端口以供外部访问

SSH

集群内可通过pod IP直接进行SSH连接（若云主机对接kube-ovn，需要在创建了ovn0设备的节点上访问）

集群外可通过公网IP（EIP），ingress-nginx暴露四层等方式映射22端口以供集群外部访问

VNC

通过 virtvnc 服务访问

远程桌面（Windows）

可通过NodePort，ingress-nginx暴露四层等方式映射3389端口以供集群外部访问

删除云主机

```
kubectl delete virtualmachines centos-net-ovn-boot-dv
```

监控服务

> kubevirt-prometheus-metrics 是一个用于将 KubeVirt 指标暴露给 Prometheus 的Service，通过在 KubeVirt 中嵌入 Prometheus 客户端库，将 KubeVirt 的指标数据暴露给 Prometheus 服务器。

> kubevirt-prometheus-metrics 通过在 KubeVirt 组件中注入 Prometheus 客户端，自动从 KubeVirt 的内部组件（如 virt-api、virt-controller、virt-handler 等）收集指标数据，并将其暴露给 Prometheus 进行数据采集。

监控使用prometheus-operator，创建ServiceMonitor将kubevirt暴露给prometheus进行监控，创建 kubevirt-servicemonitor.yml 文件， 内容如下：

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: prometheus-kubevirt-metrics
  namespace: kubevirt
  labels:
    application: prometheus-servciemonitor
spec:
  endpoints:
  - bearerTokenSecret:
      key: ""
    port: metrics
    scheme: https
    tlsConfig:
      ca: {}
      cert: {}
      insecureSkipVerify: true
  namespaceSelector:
    matchNames:
    - kubevirt
  selector:
    matchLabels:
      app.kubernetes.io/component: kubevirt
```

```
# kubectl create -f kubevirt-servicemonitor.yml
```

> 注：metadata中的labels需要与prometheus-operator中定义的LabelSelector相匹配。spec.selector.matchLabels 中的label与endpoint中的label相匹配。

创建过后在prometheus界面能够看到被发现的kubevirt相关的target，说明监控数据已被采集。

卸载服务

> 注意：卸载服务之前，需要手动移除所有VirtualMachine资源，并确认删除完成，否则会阻塞卸载

云产品：云原生云主机解除安装

过程略

> 该部分仅做内部参考

补充说明

以下功能有条件支持

功能点	说明
创建快照	依赖 VolumeSnapshot 能力，该能力并非k8s核心API，而是 CRDs。需要 CSI drivers 支持并提供这些能力。
热迁移	需要 CSI 的 external provisioner 支持 `ReadWriteMany` 能力
云主机规格配置	新版本kubevirt提供了 `VirtualMachineFlavor` ，`VirtualMachineClusterFlavor` 资源定义规格，但该规格仅支持配置cpu相关字段，可定义“通用计算型”，“计算优化型”两种规格类型，但无法达到ESCloud同等细化程度。当前官方文档还未对该特性进行说明，使用方法仅能从源码中推测。</br>之前版本已存在 `VirtualMachineInstancePreset` 资源可覆盖该部分能力，两者若同时使用推测会产生冲突。</br>另外，据官方文档说明，`VirtualMachineInstancePreset` 的设计思路（behaving）与Kubernetes原生的 `PodPresets` 一致，但 `PodPresets` 机制已在Kubernetes的新版本中被移除（详见：remove pod presets #94090 ），故有理由怀疑 `VirtualMachineInstancePreset` 可能在未来版本被弃用然后由 `VirtualMachineFlavor` ，`VirtualMachineClusterFlavor` 资源顶替。</br>

	</br>（8.12更新）根据社区最新动态，`VirtualMachineInstancePreset`确认将被弃用，当前代码中存在的`VirtualMachineFlavor`，`VirtualMachineClusterFlavor`则更名为`VirtualMachineInstanceType`，`VirtualMachinePreference`。</br>详见：PR: Deprecate VirtualMachineInstancePresets #8069
编辑镜像配置	需要 CSI 的 external provisioner 支持 `resize` 能力

以下功能不支持

功能点	说明
编辑云主机名称	该功能在v0.41.0版本后移除，详见：Removal of VM rename #5564