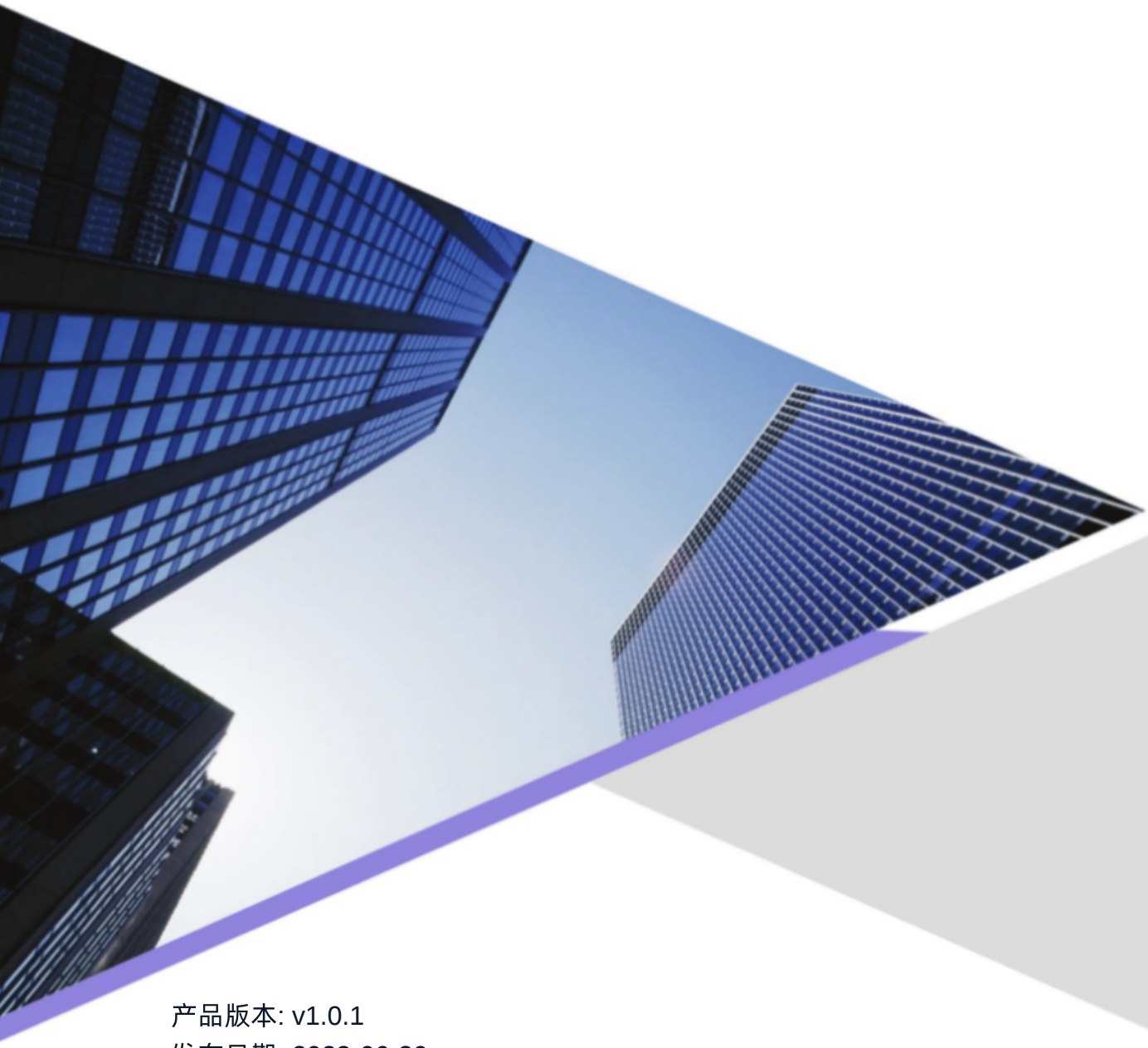
易捷行云
EasyStack

云原生云主机

快速入门



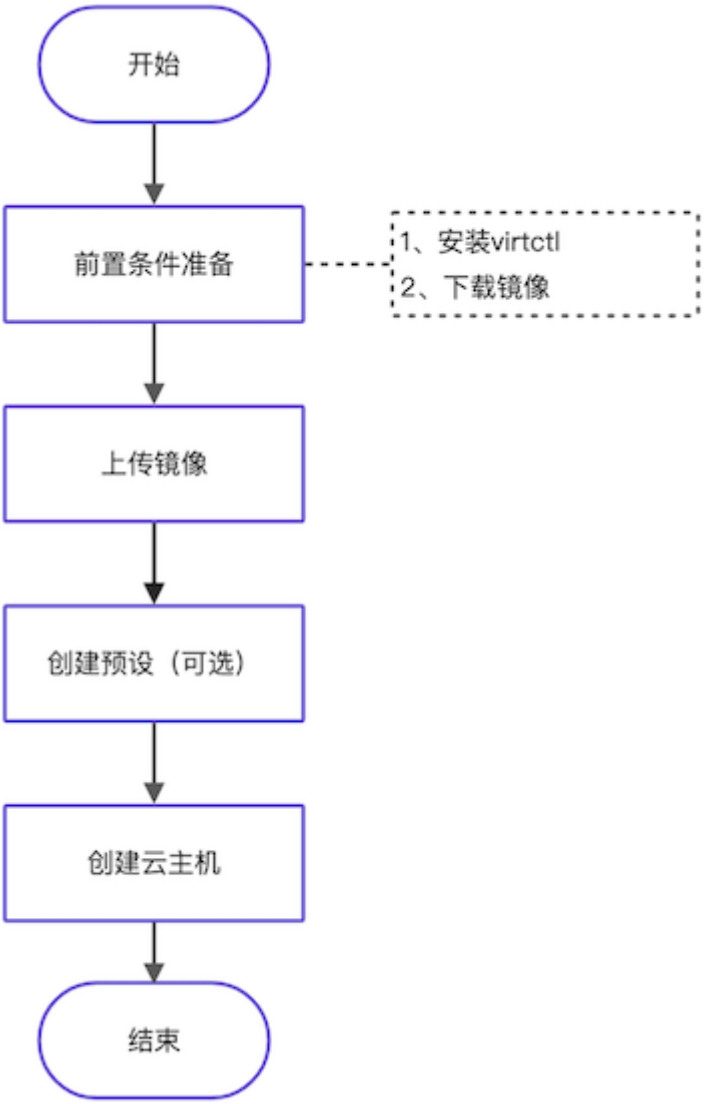
产品版本: v1.0.1
发布日期: 2023-06-20

目录

1 快速入门

1.1 操作指引

云原生云主机云产品的主线使用流程及具体说明如下：



操作流程	描述

操作流程		描述
前置条件准备	安装virtctl	安装KubeVirt自带的命令行工具virtctl，以实现越过virt-launcher Pod层直接管理云主机。
	下载镜像	下载云主机创建时所需的镜像文件。
上传镜像		上传云主机创建时所需的镜像文件。
创建预设（可选）		为云主机创建不同计算和内存能力的规格模板，供云主机创建时使用。 请根据客户实际业务需求酌情配置。如无需通过预设创建云主机，可跳过本步骤。
创建云主机		依据客户实际业务需求，创建包含CPU、内存、操作系统、网络配置、云硬盘等基础资源的云原生云主机，为客户提供可靠、安全、灵活、高效的计算环境。

1.2 前置条件准备

在使用云原生云主机服务前，请先完成以下准备工作。

安装virtctl

virtctl是KubeVirt自带的类似kubectl的命令行工具，可以越过virt-launcher Pod层直接管理云主机，如控制云主机的启动、关机和重启等。

1. 获取virtctl安装包。具体命令如下:

```
wget
https://github.com/kubevirt/kubevirt/releases/download/v0.47.1/virtctl-
v0.47.1-linux-amd64
```

2. 修改文件执行权限。具体命令如下:

```
chmod a+x virtctl-v0.47.1-linux-amd64
```

3. 复制文件至 **/usr/local/bin/virtctl** 目录。具体命令如下:

```
cp virtctl-v0.47.1-linux-amd64 /usr/local/bin/virtctl
```

下载镜像

根据客户实际业务需求，下载对应类型的镜像文件。

- CentOS镜像
- Windows镜像

1.3 上传镜像

本操作用于预先上传云主机创建时所需要的镜像文件。云原生云主机服务将通过CDI模块中的cdi-uploadproxy服务上传镜像文件。由于节点证书限制，cdi-uploadproxy服务仅支持域名访问，所以请在集群内部节点的本地，执行以下操作上传镜像。

> 说明：
> - 当需要在集群外部上传镜像时，请根据具体的网络情况配置NodePort或Ingress后，通过其进行访问。
> - 任意格式的镜像文件在上传后，都将自动转换为raw格式。

1. 配置镜像文件所在节点的hosts文件。具体命令如下:

```
kubectl get svc cdi-uploadproxy -n kubevirt | awk 'NR==2{print $3,$1}' >>
/etc/hosts
```

2. 上传镜像文件。具体命令如下:

```
virtctl image-upload pvc  --size= --image-path=<镜像地址> --storage-
class=general --wait-secs=240 --uploadproxy-url=https://cdi-uploadproxy --
insecure
```

参数	说明
PVC名称	用于保存该镜像的持久卷声明（PVC）的名称。
PVC大小	用于保存该镜像的持久卷声明（PVC）的大小。 该参数值必须大于镜像转换后的大小。
镜像地址	该镜像文件在节点中的地址。 该参数值必须包含完整的访问路径和文件名称，如： */root/kubevirt-demo/img/CentOS-7-x86_64-GenericCloud-2009.qcow2* 。

1.4 创建云主机

本操作用于创建包含CPU、内存、操作系统、网络配置、云硬盘等基础资源的云原生云主机，为客户提供可靠、安全、灵活、高效的计算环境。

1. 根据客户实际业务需求，规划云主机配置项。各配置项的具体说明及格式如下：

- CPU:

```
spec:
  template:
    spec:
    domain:
        cpu:
        cores:
```

- 内存:

```
spec:
  template:
    spec:
    domain:
        resources:
            requests:
                memory: <内存大小>G
```

- 存储：
  - 持久卷声明（PVC）：

```
spec:
  template:
    spec:
      volumes:
        - name: <数据卷名称>
            persistentVolumeClaim:
                claimName:
```

- 云硬盘（DV，DataVolume）：在云主机创建时，指定的DV会同步创建一个PVC供云主机使用。在云主机删除时，该PVC也会同步销毁:

```
spec:
  template:
    spec:
      volumes:
        - name: <数据卷名称>
            dataVolume:
                name:
```

- 云硬盘模板（DV，DataVolume）：在云主机创建时，DV模板会同步创建一个DV供云主机使用。在云主机删除时，该DV连同PVC也会同步销毁。在下述示例中，该DV模板定义的数据来源为另一个PVC，但是也支持http等多种来源:

```
spec:
  dataVolumeTemplates:
  - metadata:
        name:
    spec:
        pvc:
            accessModes:
            - ReadWriteOnce
            resources:
                requests:
                    storage: Gi
            storageClassName: <存储类名称>
        source:
            pvc:
                namespace: default
                name:
```

> 警告:
> 在同一命名空间下，请确保各DV和DV模板的名称均不重复，

- 系统盘:

```
spec:
  template:
    spec:
        domain:
            devices:
                disks:
                - disk:
                    bus: virtio
                name: <系统盘名称>
```

- 启动顺序:

```
spec:
  template:
    spec:
        domain:
            devices:
                disks:
                - bootOrder: 1
                    cdrom:
                    bus: <第一启动盘类型>
                    name: <第一启动盘名称>
                - bootOrder: 2
                    disk:
                    bus: <第二启动盘类型>
                    name: <第二启动盘名称>
```

- 网络:
  - 使用Pod默认网络:

```
spec:
  template:
    spec:
        networks:
        - name: default
          pod: {}
```

- 使用集群第二网络kube-ovn：

  当承载云原生云主机运行的是安全容器集群（即已安装安全容器服务云产品）时，可在创建云主机时，使用安全容器服务云产品默认安装的kube-ovn。在配置下述内容前，请确保已安装Multus服务，且在安全容器服务的命名空间下已配置NetworkAttachmentDefinition资源:

```
spec:
  template:
    spec:
        networks:
        - name: default
          multus:
            default: true
            networkName: secure-container/kube-ovn
```

  当使用集群第二网络kube-ovn时，接口设置仅支持bridge模式。此外，集群第二网络kube-ovn需配合固定IP地址使用，否则会由于kube-ovn的内置IPAM分配问题，导致云主机在重启后会网络异常（其中，mac_address请自动生成，不可复用已使用的mac地址）：

```
spec:
  template:
    metadata:
        annotations:
            ovn.kubernetes.io/ip_address: <固定IP地址>
            ovn.kubernetes.io/mac_address: 00:00:00:63:D5:F9
```

- 接口：

  - masquerade模式（默认）：

```
  spec:
    template:
      spec:
          domain:
              devices:
                  interfaces:
                  - masquerade: {}
                  name: default
```

- bridge模式:

```
spec:
  template:
    spec:
        domain:
            devices:
                interfaces:
                - bridge: {}
                name: default
```

> 警告:
> - 当使用集群第二网络kube-ovn时，仅支持bridge模式。
> - 当使用Pod默认网络对接bridge模式时，请在KubeVirt提供的permitBridgeInterfaceOnPodNetwork配置项中禁用该关联，否则云主机将无法成功创建。

- 密钥/密码注入:

```
spec:
  template:
    spec:
        domain:
            devices:
                disks:
                - disk:
                bus: virtio
                name: cloudinitdisk
        volumes:
            - cloudInitNoCloud:
                userData: |
                    #cloud-config
                    disable_root: false
                    ssh_pwauth: true
                    ssh_authorized_keys:
                    - ''
                    users:
                    - name: escore
                        gecos: ES Core User
```

```
                            sudo: ALL=(ALL) NOPASSWD:ALL
                            passwd:
                            shell: /bin/bash
                            home: /home/escore
                            lock_passwd: false
                            ssh_pwauth: true
                    name: cloudinitdisk
```

2. 通过Yaml，创建云主机。

下文将分别列举几个Linux和Windows操作系统云主机的Yaml文件格式供参考。

- Linux云主机
  - 使用默认Pod网络，并通过存有镜像文件的PVC启动:

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
    name: <云主机名称>
spec:
    running: true
    template:
    metadata:
        labels:
            kubevirt.io/domain: <云主机名称>
    spec:
        domain:
            cpu:
                cores:
            devices:
                disks:
                - disk:
                bus: virtio
                name: boot-disk
                interfaces:
                - masquerade: {}
                name: default
            machine:
                type: q35
            resources:
```

```
                    requests:
                    memory: <内存大小>G
            networks:
                - name: default
                pod: {}
            volumes:
                - name: <数据卷名称>
                persistentVolumeClaim:
                    claimName:
```

- 使用集群第二网络，并设置为固定IP地址，且通过运硬盘模板定义的运硬盘启动，此外另配置 **cloudInitNoCloud** 卷用于注入密码与秘钥:

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
    name: <云主机名称>
spec:
    running: true
    template:
    metadata:
        annotations:
            ovn.kubernetes.io/ip_address: 10.16.1.110
            ovn.kubernetes.io/mac_address: 00:00:00:63:D5:F9
        labels:
            kubevirt.io/domain: <云主机名称>
    spec:
        domain:
            cpu:
                cores:
            devices:
                disks:
                - disk:
                bus: virtio
                name: boot-disk
                - disk:
                bus: virtio
                name: cloudinitdisk
                interfaces:
                - bridge: {}
```

```
                name: default
            machine:
                type: q35
            resources:
                requests:
                memory: <内存大小>G
        networks:
            - name: default
            multus:
                default: true
                networkName: secure-container/kube-ovn
        volumes:
            - name: <数据卷名称>
                dataVolume:
                    name:
            - cloudInitNoCloud:
                userData: |
                    #cloud-config
                    disable_root: false
                    ssh_pwauth: true
                    ssh_authorized_keys:
                    - ''
                    users:
                    - name: escore
                        gecos: ES Core User
                        sudo: ALL=(ALL) NOPASSWD:ALL
                        passwd:
                        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: Gi
            storageClassName: <存储类名称>
        source:
            pvc:
                namespace: default
                name:
```

- Windows云主机
  - iso启动：

    1. 通过Yaml，创建持久卷声明（PVC）。Yaml文件格式如下:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
    name:
spec:
    accessModes:
        - ReadWriteOnce
    resources:
        requests:
            storage: Gi
    storageClassName: <存储类名称>
```

    2. 通过Yaml，创建云主机。Yaml文件格式如下(其中， *hub.example.io/production/virtio-container-disk* 为包含Windows云主机所需驱动的一个容器盘。):

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
name: <云主机名称>
spec:
running: true
template:
    metadata:
        labels:
            kubevirt.io/domain: <云主机名称>
    spec:
```

```
        domain:
            cpu:
            cores:
            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: <内存大小>G
        networks:
            - name: default
            pod: {}
        volumes:
            - name: <数据卷名称>
            persistentVolumeClaim:
                claimName:
            - name: <数据卷名称>
            persistentVolumeClaim:
                claimName:
            - containerDisk:
                image: hub.example.io/production/virtio-container-disk
            name: virtiocontainerdisk
```

- raw启动:

```
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:
            devices:
                disks:
                - disk:
                bus: virtio
                name: boot-disk
                interfaces:
                - masquerade: {}
                name: default
            machine:
                type: q35
            resources:
                requests:
                    memory: <内存大小>G
        networks:
            - name: default
                pod: {}
        volumes:
            - name: <数据卷名称>
                dataVolume:
                    name:
    dataVolumeTemplates:
    - metadata:
    name: win2k19-dv
    spec:
    pvc:
        accessModes:
        - ReadWriteOnce
```

```
            resources:
                requests:
                    storage: Gi
            storageClassName: <存储类名称>
        source:
            pvc:
                namespace: default
                name:
```

咨询热线：400-100-3070

北京易捷思达科技发展有限公司：
北京市海淀区西北旺东路10号院东区1号楼1层107-2号
南京易捷思达软件科技有限公司：
江苏省南京市雨花台区软件大道168号润和创智中心4栋109-110

邮箱：
contact@easystack.cn (业务咨询)
partners@easystack.cn(合作伙伴咨询)
marketing@easystack.cn (市场合作)