

# 鼎甲迪备

## Kubernetes 备份恢复用户指南

Release V8.0-7

August, 2023



目录	i
表格索引	iii
1 概述	1
2 计划和准备	3
3 代理端安装和配置	5
3.1 验证兼容性	5
3.2 安装迪备代理端	5
3.2.1 Linux 操作系统	5
3.3 检查代理端安装成功	8
4 激活许可证和授权用户	9
5 备份	11
5.1 备份类型	11
5.2 备份策略	11
5.3 开始之前	11
5.4 登录实例	12
5.5 环境检查	12
5.6 创建备份作业	14
6 恢复	17
6.1 开始之前	17
6.2 创建时间点恢复作业	17
6.3 恢复选项	18
7 限制性	19
8 术语表	21



1	备份常规选项列表 . . . . .	15
2	备份高级选项列表 . . . . .	15
3	恢复高级选项 . . . . .	18
4	限制性列表 . . . . .	19
5	术语表 . . . . .	21

该文档主要描述了如何安装配置迪备代理以及如何正确使用迪备备份和恢复 Kubernetes。

迪备支持 Kubernetes 备份恢复主要特性包括：

- 备份内容

Namespace、Workloads

- 备份类型

完全备份

- 备份目标

标准存储池、重删存储池、磁带库池、对象存储池、LAN-Free 池

- 备份策略

迪备提供 6 种备份计划，立即、一次、每小时、每天、每周、每月

- 数据处理

数据压缩、数据加密、多通道、断点续传、限速、复制

- 恢复类型

时间点恢复

- 恢复目标

原机、异机

- **Kubernetes 备份**：备份是为已有的 Kubernetes 卷创建副本，它可以像任何其它标准卷一样被使用。唯一的区别就是配置后，后端设备将创建指定完全相同的副本，而不是创建一个“新的”空卷。备份功能会保存 Kubernetes 的云原生资源数据以及对克隆卷内的数据进行备份，以保障 Kubernetes 资源数据与应用持久化数据的一致性。



在安装迪备代理端之前，请确保满足以下要求：

1. 确保所有备份组件都已安装和部署，包括备份服务器、存储服务器。
2. 准备一个至少具备操作员和管理员角色的用户，使用此用户登录迪备控制台进行后续操作。

---

**备注：** 管理员角色用于代理端安装和配置、激活许可证和授权用户。操作员角色用于创建备份和恢复作业。

---





## 3.1 验证兼容性

在安装代理之前，请先确保 Kubernetes 所在主机环境已在鼎甲迪备的适配列表中。

- Kubernetes 1.17/1.18/1.19/1.20/1.21/1.22/1.23/1.26

## 3.2 安装迪备代理端

迪备支持 Linux 平台上的 Kubernetes 集群备份恢复。

### 3.2.1 Linux 操作系统

Kubernetes 通过镜像的方式下载安装包进行安装。以下操作都在 master 节点执行。

1. 如不能在 master 节点操作。则需要从 master 节点拷贝 admin.conf 文件到 node 节点上：

```
scp /etc/kubernetes/admin.conf root@<节点 IP>:/etc/kubernetes/  
echo "export KUBECONFIG=/etc/kubernetes/admin.conf" >> ~/.bash_profile  
source ~/.bash_profile
```

2. 下载镜像（每个节点都需要下载）  
镜像名称：

```
docker load -i agent-k8s-version.tar
```

3. 查看镜像

```
sudo docker images | grep k8s 或用 ctr images ls | grep k8s 命令查看
```

4. 创建命名空间和指定 hostid

```
kubect1 create ns backup  
uuidgen -r | sed "s/-//g" # configmaps 中 hostid 生成需要自己指定
```

5. cluster.yaml 参考配置如下

```

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: backup-k8s
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: default
  namespace: backup

kubect1 apply -f cluster.yaml

```

6. agent 参考配置如下（可部署多代理执行多作业，适应代理端在各节点可切换，部署多个代理端时可能调度到同一节点）

```

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: backup-agent
  namespace: backup
spec:
  selector:
    matchLabels:
      app: backup-agent
  serviceName: backup-agent
  replicas: 1 # 副本数
  template:
    metadata:
      labels:
        app: backup-agent
    spec:
      containers:
        - env:
            - name: BACKUPD_HOST
              value: 172.16.30.197
            - name: BACKUPD_PORT
              value: "50305"
            - name: HOSTID_0 # 指定hostid, 从编号0开始
              value: 9e26580745ab41fea2f80bf96e739186
            - name: BACKUPD_SSL
              value: "false"
            - name: POD_IMAGE
              # 根据实际情况修改, docker images | grep pause
              value: registry.aliyuncs.com/k8sxio/pause:3.2
            - name: POD_NAME
              value: backup-pod
            - name: DEPLOY_METHOD
              value: statefulset
            - name: HOSTNAME
              valueFrom:

```

(续下页)

(接上页)

```

        fieldRef:
          fieldPath: metadata.name
# - name: BACKUPD_ACCESS_KEY
#   value: "42fff1271225cf15198e55a886e78945"
- name: BACKUP_NODE
  valueFrom:
    fieldRef:
      fieldPath: spec.nodeName
# 镜像名，根据实际情况进行修改
image: registry.docker.scutech.com/stable/focal/agent-k8s:version
imagePullPolicy: IfNotPresent
name: agent
resources: {}
securityContext:
  privileged: true
volumeMounts:
  - mountPath: /var/log/dbackup3
    name: log-volume
  - mountPath: /var/opt/scutech/dbackup3/agent
    name: opt-volume
  - mountPath: /var/lib/kubelet/pods
    mountPropagation: HostToContainer
    name: pods-path
  - mountPath: /dev
    mountPropagation: HostToContainer
    name: dev
hostIPC: true
hostNetwork: true
hostPID: true
#nodeName: k8s-master-85 #指定node节点，LAN-Free池必须配置，默认可不配置
#nodeSelector:
#   kubernetes.io/hostname: k8s-master-85 #指定node节点，LAN-Free池必须配置
volumes:
  - hostPath:
      path: /var/lib/kubelet/pods
      name: pods-path
  - hostPath:
      path: /dev
      name: dev
  - hostPath:
      path: /opt/data/opt_volume
      name: opt-volume
  - hostPath:
      path: /opt/data/log_volume
      name: log-volume

```

### 3.3 检查代理端安装成功

1. 查看代理端资源是否为 running，`kubectl get pod -n backup`（backup 为命名空间，可根据实际情况查看）

```
[root@k8s-master-106 ~]# kubectl get pod -n backup
NAME             READY   STATUS    RESTARTS   AGE
backup-agent-0    1/1     Running   3           7h46m
```

## 激活许可证和授权用户

代理端安装成功后，返回迪备控制台**【资源】**页面，列表中会出现安装了代理端的主机。在备份恢复之前，您需要在迪备控制台对代理端的 Kubernetes 进行激活许可和授权用户。

1. 管理员登录迪备控制台。
2. 进去**【资源】**页面，您可以通过**【搜索】**找到 Kubernetes 主机，点击**【注册】**。
3. 弹出**【激活】**窗口。勾选 Kubernetes 备份的许可模块。点击**【提交】**。

---

**备注：**若提示“许可证不足”，请联系迪备管理员增加许可证。

---

4. 在**【授权】**窗口中，选择当前控制台用户所在的用户组，点击**【提交】**。

---

**备注：**若代理端数量较多，建议对所有代理端先完成代理端安装，再使用**【批量注册】**、**【批量激活】**和**【批量授权】**，以减少操作次数。具体查看管理员手册的《批量注册/激活/授权》。

---



## 5.1 备份类型

迪备为 Kubernetes 备份提供了一种常规的备份类型：

- 完全备份

备份 Kubernetes 中的所有数据，以及可以恢复这些数据的足够日志。

## 5.2 备份策略

迪备提供 6 种备份计划，立即、一次、每小时、每天、每周、每月。

- 立即：作业创建后就执行。
- 一次：作业在指定时间执行一次。
- 每小时：作业每天在设置的时间范围内以特定的小时/分钟间隔重复运行。
- 每天：作业以特定的天数间隔在特定时间重复运行。
- 每周：作业以特定的周数间隔在特定时间重复运行。
- 每月：作业在特定月份和时间重复运行。

通常，推荐用户使用常规的备份策略：

1. 完全备份：每周在应用访问量比较小的时候，进行一次**完全备份**。保证每周至少有个可恢复的 RTO。

## 5.3 开始之前

在备份恢复 Kubernetes 集群之前，请保证已完成如下操作：

1. 检查资源状态

操作员登录迪备控制台，进入【资源】页面，主机和 Kubernetes 资源都显示“在线”状态。如果为离线状态，请检查迪备代理端服务、Kubernetes 服务是否正常运行。

2. 检查存储池

(1) 在迪备菜单栏中，点击【存储池】，进入【存储池】页面。

(2) 检查展示区是否存在存储池。如果没有，请参考管理手册《存储池》，创建存储池并授权给当前控制台用户。

## 5.4 登录实例

创建备份恢复作业之前。您必须先在必备控制台上登录 Kubernetes 实例，对 Kubernetes 做身份验证。必备支持一种身份认证方式：

- Access key 认证

使用当前必备用户的 Access Key 身份验证登录。适用于无法获取操作系统用户密码或用户密码频繁变更的场景。

备注：

1. Access Key 认证默认未启用。若要开启，需登录必备控制台，进入【设置】页面，打开【安全】标签页，勾选【Access Key 登录实例】。
2. 获取用户 Access key：登录控制台，点击右上角【个人设置】，选择【账号设置】，在【首选项】找到 Access Key，并点击【查看】，获取当前登录用户的 Access Key。

## 5.5 环境检查

Kubernetes 备份恢复之前，请检查代理端上的 Kubernetes 集群状态是否正常。这里提供了 Linux 平台上检查 Kubernetes 集群状态的命令。

1. `systemctl status kubelet`，查看 kubelet 服务是否为正常 active(running)。

```
[root@k8s-master-106 ~]# systemctl status kubelet
• kubelet.service - kubelet: The Kubernetes Node Agent
  Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; vendor preset: ❏
  ↳ disabled)
  Drop-In: /usr/lib/systemd/system/kubelet.service.d
           └─10-kubeadm.conf
  Active: active (running) since Thu 2022-10-20 16:47:15 CST; 53min ago
  Docs: https://kubernetes.io/docs/
  Main PID: 1169 (kubelet)
  Tasks: 29
  Memory: 161.6M
  CGroup: /system.slice/kubelet.service
           └─1169 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-
  ↳ kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/
  ↳ config.yaml --network-plugin...
```

2. `kubectl get nodes -owide`，查看 Kubernetes 集群的部署版本，请确保所有节点的 version 不低于 v1.17.0，否则不支持 CSI 驱动，并查看所有 node 节点是否为 Ready。

```
[root@k8s-master-106 ~]# kubectl get nodes -owide
NAME                STATUS  ROLES    AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-
↳ IMAGE              KERNEL-VERSION   CONTAINER-RUNTIME
k8s-master-106     Ready   master   30d   v1.19.5   172.16.12.106 <none>         ❏
↳ CentOS Linux 7 (Core) 5.4.219-1.el7.elrepo.x86_64 docker://19.3.11
k8s-node-107       Ready   <none>   30d   v1.19.5   172.16.12.107 <none>         ❏
↳ CentOS Linux 7 (Core) 5.4.219-1.el7.elrepo.x86_64 docker://19.3.11
k8s-node-108       Ready   <none>   30d   v1.19.5   172.16.12.108 <none>         ❏
```

(续下页)



(接上页)

```
↪CentOS Linux 7 (Core) 5.4.219-1.el7.elrepo.x86_64 docker://19.3.11
```

3. `kubectl get pod -A`, 查看 Kubernetes 集群所有 pod 是否为 Running。

```
[root@k8s-master-106 ~]# kubectl get pod -A
```

NAMESPACE	NAME	READY	STATUS		
↪RESTARTS	AGE				
backup	backup-agent-0	1/1	Running	3	↪
↪	7h15m				
kube-system	calico-kube-controllers-6c89d944d5-sgspg	1/1	Running	9	↪
↪	30d				
kube-system	calico-node-442p9	1/1	Running	9	↪
↪	30d				
kube-system	calico-node-hqlx4	1/1	Running	7	↪
↪	30d				
kube-system	calico-node-srnvz	1/1	Running	7	↪
↪	30d				
kube-system	coredns-59c898cd69-bcf7r	1/1	Running	9	↪
↪	30d				
kube-system	coredns-59c898cd69-t97vc	1/1	Running	10	↪
↪	30d				
kube-system	etcd-k8s-master-106	1/1	Running	9	↪
↪	30d				
kube-system	kube-apiserver-k8s-master-106	1/1	Running	13	↪
↪	30d				
kube-system	kube-controller-manager-k8s-master-106	1/1	Running	76	↪
↪	30d				

4. `kubectl get sc`, 查看是否存在 CSI 驱动对应的 StorageClass, 以及 StorageClass 对应的 RECLAIMPOLICY 是否为 Delete; 若 RECLAIMPOLICY 不为 Delete, 则在对该资源进行备份后将无法删除备份临时创建出来的 PV 和 Ceph 中的卷。

```
[root@k8s-master-106 ~]# kubectl get sc
```

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE	
↪ALLOWVOLUMEEXPANSION	AGE			
csi-rbd-sc	rbd.csi.ceph.com	Delete	Immediate	true ↪
↪	30d			

5. `ceph -s`, 查看 ceph 集群状态是否正常, 备份前确保 ceph 集群有足够空间, ceph 版本: ceph version 14.2.22 (ceph 版本需求大于 v14.0), 并且确保内核版本不低于 5.1。

```
[root@k8s-master-106 ~]# ceph -s
```

```
cluster:
  id:          948d9908-dd20-4866-beea-e798e82f0252
  health: HEALTH_OK

services:
  mon: 1 daemons, quorum k8s-master-106 (age 24h)
  mgr: k8s-master-106(active, since 24h)
  osd: 3 osds: 3 up (since 24h), 3 in (since 5d)

data:
  pools:   1 pools, 128 pgs
```

(续下页)

```
objects: 825 objects, 2.3 GiB
usage:   9.8 GiB used, 590 GiB / 600 GiB avail
pgs:     128 active+clean

io:
  client: 62 KiB/s wr, 0 op/s rd, 4 op/s wr
```

#### 备注:

1. 需满足所有环境检查条件才能进行 Kubernetes 备份。
2. Kubernetes v1.20 开始，默认删除了 `metadata.selfLink` 字段，然而部分应用仍然依赖于这个字段，例如 `nfs-client-provisioner`。如果仍然要继续使用这些应用，您将需要重新启用该字段。请修改 `/etc/kubernetes/manifests/kube-apiserver.yaml` 文件，并在其启动参数中增加一行 `--feature-gates=RemoveSelfLink=false` 并执行 `kubectl apply -f kube-apiserver.yaml`
3. 备份恢复到 LAN-Free 池，需要执行以下步骤。第一步：所有宿主机 load 内核 `modprobe iscsi_tcp`，可通过 `lsmod | grep iscsi_tcp` 命令查看是否执行成功。第二步：进入代理端 pod 里面执行 `/usr/sbin/iscsid`
4. 备份前确保 Ceph 集群有足够空间。
5. CSI 参考链接：[ceph-csi.git](https://github.com/ceph/ceph-csi)
6. 快照参考链接：[external-snapshotter.git](https://github.com/external-snapshotter/external-snapshotter)

## 5.6 创建备份作业

1. 在菜单栏中，点击【**备份**】，选择需要备份的主机，再选择资源中的 Kubernetes 实例，点击【**下一步**】。
2. 在【**备份内容**】页面，选择一个【**备份类型**】，勾选您希望备份的 Kubernetes，点击【**下一步**】。
3. 在【**备份目标**】页面，选择一个存储池，点击【**下一步**】。
4. 在【**备份计划**】页面，选择一个计划类型，参考[备份策略](#)，点击【**下一步**】。
  - 立即：指作业立即执行，作业提交后作业立即开始执行。
  - 一次：指作业指定执行时间，作业提交后作业处于空闲状态，等到达指定执行时间后作业开始执行。
  - 每小时：指作业根据设置的小时数，每隔小时执行作业。数值范围为 1~24 之间的整数。
  - 每天：指作业根据设置的天数，每隔天数执行作业。数值范围为 1~5 之间的整数。
  - 每周：指作业根据设置的周数，指定在每隔周数执行作业。还可设置星期数，指定这周内所选的星期几都执行一次。
  - 每月：指作业根据设置的月数，指定在每隔月数执行作业。还可设置星期/日期，指定这月内所选的星期/日期都执行一次。
5. 设置【**备份选项**】，根据需要设置常规选项和高级选项，参看备份选项。点击【**下一步**】。
6. 在【**完成**】页面，设置【**作业名**】，并检查作业信息是否有误。点击【**提交**】。
7. 提交成功后，自动跳转到作业页面。您还可以对作业进行开始、编辑、删除等管理操作。
  - 常规选项

表 1: 备份常规选项列表

功能	描述	限制性说明
压缩	默认启用快速压缩。 - 不压缩：备份过程中不压缩。 - 可调节：自定义压缩级别，需激活高级功能。 - 快速压缩：备份过程中压缩，使用快速压缩算法。	
通道数	开启该选项可提高备份效率。通道数默认为 1，选择范围为 1-255，单位为个。 一般建议跟 CPU 核心数一致，超过 CPU 核心数之后效率提高不明显。	

- 高级选项

表 2: 备份高级选项列表

功能	描述	限制性说明
断线重连时间	支持 1~60，单位为分钟。在设置时间内网络发生异常复位后作业继续进行。	
速度限制	限制数据传输速度或磁盘读写速度。单位为 MiB/s。	
前置条件	作业开始前调用，当前置条件不成立时中止作业执行，作业变成空闲状态。	
前置/后置脚本	前置脚本在作业开始后资源进行备份或恢复前调用，后置脚本在资源进行备份或恢复后调用。	



迪备提供了 Kubernetes 的恢复方式，包括：

- 时间点恢复

当 Kubernetes 发生灾难时，可以通过时间点恢复将 Kubernetes 恢复到指定时间点，并支持本机或异机恢复。

## 6.1 开始之前

如果要恢复到其他主机的 Kubernetes 实例，请先在该集群上安装代理，激活许可证，并将 Kubernetes 资源授权给当前迪备控制台用户。

## 6.2 创建时间点恢复作业

创建时间点恢复作业的步骤如下：

1. 在菜单栏中，点击 **【恢复】**，进入 **【恢复】** 页面。
2. 在 **【主机和资源】** 页面，选择神通所在主机和实例，点击 **【下一步】**。
3. 在 **【备份集】** 页面中，完成以下操作：
  - (1) **【恢复类型】** 选择**时间点恢复**。
  - (2) 选择恢复的 **【备份集】**。
  - (3) 恢复信息设置完成，点击 **【下一步】**。
4. 在 **【恢复目标】** 页面，支持恢复到本机或异机。点击 **【下一步】**。
5. 在 **【恢复计划】** 页面，选择“立即”或“一次”，点击 **【下一步】**。
  - 选择“立即”，作业创建后就执行。
  - 请选择“一次”，设置作业的开始时间。
6. 在 **【恢复选项】** 页面，参考[恢复选项](#)，根据所需进行设置。点击 **【下一步】**。
7. 在 **【完成】** 页面，设置作业名称，并确认恢复内容。点击 **【提交】**，等待作业执行。
8. 提交成功后，自动跳转到作业页面。您还可以对作业进行开始、编辑、删除等管理操作。

---

**备注：**恢复过程中，恢复作业可能因为原 service 关联配置资源存在冲突（例如 IP 或端口等情况）而导致作业失败但数据会恢复成功，需要根据报错信息手动创建 service 资源并更改 IP 和端口。

---

### 6.3 恢复选项

迪备为 Kubernetes 提供了以下恢复选项：

- 高级选项：

表 3: 恢复高级选项

功能	描述	限制性说明
断线重连时间	支持 1~60，单位为分钟。在设置时间内网络发生异常复位后作业继续进行。	
断点续传缓冲区	默认为 10MiB。设置断点续传缓冲区大小。	
速度限制	可分时段限制数据传输速度或磁盘读写速度。单位为 KiB/s、MiB/s 或 GiB/s。	
前置条件	作业开始前调用，当前置条件不成立时中止作业执行，作业变成空闲状态。	
前置/后置脚本	前置脚本在作业开始后资源进行恢复前调用，后置脚本在资源进行恢复后调用。	

表 4: 限制性列表

功能	限制描述
备份	<ol style="list-style-type: none"><li>1. 不支持对块设备的挂载卷进行备份恢复。</li><li>2. 仅支持对使用 Ceph-CSI/CephFS-CSI 持久卷的工作负载进行。</li><li>3. 不支持以集群为粒度备份。</li><li>4. 不支持 NFS 文件系统</li></ol>
环境	<ol style="list-style-type: none"><li>1. Ceph 集群环境内核大于 5.1</li><li>2. Kubernetes Version 不低于 v1.17.0。</li><li>3. Ceph 版本需求大于 v14.0。</li><li>4. 不支持 IPv6 双栈。</li><li>5. LAN-Free 存储池中恢复（支持 iSCSI）</li></ol>
恢复	<ol style="list-style-type: none"><li>1. 不能恢复 Ceph-CSI 驱动。</li><li>2. 不能恢复代理端相关 pod。</li></ol>





表 5: 术语表

术语	说明
CustomResourceDefinition	简称 CRD，本身只是一段声明，用于定义用户自定义的资源对象。
NameSpace	Kubernetes（简称 K8S）支持多个虚拟集群，它们底层依赖于同一个物理集群，这些虚拟集群被称为 <b>NameSpace</b> （命名空间，简称 NS）。通过将资源分配给不同 <b>NameSpace</b> ，可以让资源形成逻辑上的隔离，以方便对不同资源的隔离使用和管理。同一个命名空间下各种资源不能重名，不同命名空间下的资源则允许重名。
Master	K8S 的主控组件，对应的对象是 <b>Node</b> 。
Node	K8S 集群的机器节点，一个 <b>Node</b> 对应一个具体的物理机或者虚拟机。
Container	镜像容器。
Pod	可以在 K8S 中创建和管理的、最小的可部署的计算单元。一个 <b>Pod</b> 可以包含一个或者多个 <b>Container</b> 。
Service	多个相同的 <b>Pod</b> 组成一个服务，统一对外提供服务。
NodePort	支持 K8S 集群外主机与集群通讯的网络服务类型。
StorageClass	为管理员提供了一种在配置卷时描述存储“类”的方法。
VolumeSnapshotClass	提供了一种在配置卷快照时描述存储“类”的方法。
VolumeSnapshotContent	是一种快照，从管理员已提供的集群中的卷获取。就像持久卷是集群的资源一样，它也是集群中的资源。
VolumeSnapshot	是用户对于卷的快照的请求，它类似于持久卷声明。
PersistentVolume	持久卷，简称 PV，是集群内，由管理员提供的网络存储的一部分。
PersistentVolumeClaim	持久卷声明，简称 PVC，用户对存储的一种请求。
Secret	对象类型用来保存敏感信息，例如密码、OAuth 令牌和 SSH 密钥。
ConfigMap	一种 API 对象，用来将非机密性的数据保存到键值对中。
ServiceAccount	为 Pod 中运行的进程提供了一个标识。

续下页

表 5 - 接上页

术语	说明
LimitRange	LimitRange 是在命名空间内限制资源分配（给多个 Pod 或 Container）的策略对象。
CSI	CSI 全称是 Container Storage Interface。它是由来自 Kubernetes、Mesos、Docker 等社区成员联合制定的一个行业标准接口规范，旨在将任意存储系统暴露给容器化应用程序。CSI 规范定义了存储提供商实现 CSI 兼容的 Volume Plugin 的最小操作集和部署建议。
克隆	克隆（Clone）意思是为已有的 Kubernetes 卷创建副本，它可以像任何其它标准卷一样被使用。克隆只支持 CSI 驱动程序。
ReplicationController	ReplicationController 简称 RC，主要作用是创建和管理一个 Pod 的多个副本（Replicas）。
Replicaset	Replicaset 简称 RS，主要作用是维护一组在任何时候都处于运行状态的 Pod 副本的稳定集合。属于新一代的 RC，可完全替代 RC。
Deployment	Deployment 可以看作是 RC 的超集，除了提供 Pod 管理等功能，额外提供了回滚、版本记录等新特性。一般情况下我们不会直接创建 RC/RS，而是通过创建更高层级的 Deployment 资源来自动创建 RC/RS。
ApiServer	ApiServer 提供了 K8S 各类资源对象（Pod、RC、Service 等）的增删改查及监控等 HTTP Rest 接口，是整个系统的数据总线 and 数据中心。
元数据	该文档用元数据表示 Namespace 中除持久卷存储数据外的其它资源数据。
DaemonSet	DemonSet 能确保其创建的 Pod 在集群中的每一台（或指定）Node 上都运行一个副本。如果集群中动态加入了新的 Node，DaemonSet 中的 Pod 也会被添加在新加入 Node 上运行。删除一个 DaemonSet 也会级联删除所有其创建的 Pod。
StatefulSet	StatefulSet 是一个给 Pod 提供唯一标志的控制器，它可以保证部署和扩展的顺序。
kubectI	kubectI 是 Kubernetes 的命令行工具（CLI），是 Kubernetes 用户和管理员必备的管理工具。