# Kubernetes Etcd 数据管理

前言：

课程名称：Kubernetes Etcd 数据管理

实验环境：

本章节 Kubernetes 集群环境如下：

| 角色 | IP | 主机名 | 组件 | 硬件 |
| --- | --- | --- | --- | --- |
| 控制节点 | 192.168.128.11 | k8s-master01 | apiserver<br>controller-manager<br>scheduler<br>etcd<br>containerd | CPU：4vCPU<br>硬盘：100G<br>内存：4GB<br>开启虚拟化 |
| 工作节点 | 192.168.128.21 | k8s-node01 | kubelet<br>kube-proxy<br>containerd<br>calico<br>coredns | CPU：6vCPU<br>硬盘：100G<br>内存：6GB<br>开启虚拟化 |

张岩峰老师微信，加我微信，邀请你加入 VIP 交流答疑群：

微信号：ZhangYanFeng0429

二维码：

## 1、Etcd 在 Kubernetes 集群中的作用

etcd 在 Kubernetes 集群中扮演着关键的角色。它是一个分布式键值存储系统，用于存储 Kubernetes 集群的配置数据和状态信息。etcd 提供了高可用性和一致性，确保集群中的各个组件可以共享和同步数据。

etcd 被用于存储 Kubernetes 的各种资源对象，如 Pod、Service、ConfigMap 等。这些对象的配置信息和状态变化都会被写入 etcd 中，并且可以被集群中的其他组件读取和监听。

etcd 的作用不仅限于存储数据，它还负责维护集群的一致性和可用性。当集群中的节点发生故障或网络分区时，etcd 可以自动进行故障检测和恢复，确保集群的正常运行。

总之，etcd 在 Kubernetes 集群中扮演着数据存储和同步的关键角色，为集群的可靠性和高可用性提供了基础支持。

## 2、Etcd 集群选举算法

1、初始启动时，节点处于 follower 状态并被设定一个 election timeout，如果在这一时间周期内没有收到来自 leader 的 heartbeat，节点将发起选举：将自己切换为 candidate 之后，向集群中其它 follower 节点发送请求，询问其是否选举自己成为 leader。

2、当收到来自集群中过半数节点的接受投票后，节点即成为 leader，开始接收保存 client 的数据并向其它的 follower 节点同步日志。如果没有达成一致，则 candidate 随机选择一个等待间隔（150ms ~ 300ms）再次发起投票，得到集群中半数以上 follower 接受的 candidate 将成为 leader。

3、leader 节点依靠定时向 follower 发送 heartbeat 来保持其地位。

4、任何时候如果其它 follower 在 election timeout 期间都没有收到来自 leader 的 heartbeat，同样会将自己的状态切换为 candidate 并发起选举。每成功选举一次，新 leader 的任期（Term）都会比之前 leader 的任期大 1。

## 3、Etcd 基本操作

说明：这里是以 kubeadm 部署的 k8s 集群，etcd 服务是以 pod 容器方式存在的。

### 3.1、Etcd pod 内的基本操作

#### 3.1.1、获取 etcd 成员列表和 etcd 节点状态

etcd 的 API 分为两种，分别用 export ETCDCTL_API=3 和 export

ETCDCTL_API=2 来区分。两种 API 的调用接口不同，其数据组织形式也不同。在 API_2 下，其 key 和 value 都存储在内存中，而在 API_3 下，key 存储在内存中，value 存储在硬盘中。

（1）查看 etcd pod 并进入 Pod 内

```
1、查看 etcd 集群中运行的 etcd pod
    [root@k8s-master01 ~]# kubectl get pods -n kube-system | grep etcd
    etcd-k8s-master01          1/1     Running   2 (5m48s ago)   102d

2、进入 etcd pod 容器
    [root@k8s-master01 ~]# kubectl exec -it etcd-k8s-master01 -n kube-system --
/bin/sh

3、设置环境变量为 v3
    sh-5.0# export ETCDCTL_API=3

说明：为什么要设置环境变量为 v3?
    Kubernetes 会使用 etcd v3 版本的 API 记录数据。而默认 etcdctl 是使用 v2 版本的 API，查看不到 v3 的数据。设置环境变量 ETCDCTL_API=3 后就可以查到 v3 版本的 API 记录的数据。
```

2、获取 etcd 的成员列表

（table 形式）也可以使用 json（-w json）或者普通显示（member list）

（1）获取 etcd 的成员列表，以表格形式显示。

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 -w table member list
```

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 -w table member list
+------------------+---------+-------------+---------------------------+---------------------------+------------+
|        ID        | STATUS  |     NAME    |         PEER ADDRS        |        CLIENT ADDRS       | IS LEARNER |
+------------------+---------+-------------+---------------------------+---------------------------+------------+
| efc91ac160b34f04 | started | k8s-master01 | https://192.168.128.11:2380 | https://192.168.128.11:2379 |      false |
+------------------+---------+-------------+---------------------------+---------------------------+------------+
sh-5.1#
```

（2）获取 etcd 的成员列表，普通显示。

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 member list
```

**版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。**

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 member list
efc91ac160b34f04, started, k8s-master01, https://192.168.128.11:2380, https://192.168.128.11:2379, false
sh-5.1#
```

对上面参数说明：

etcdctl 是用于与 etcd 数据库进行交互的工具。此命令将列出 etcd 集群的成员。

以下是这个命令的解释：

--cacert：这个参数指定了 etcd 集群的 CA 证书的路径。这个证书用于验证 etcd 服务器证书的合法性。

--cert：这个参数指定了 etcdctl 客户端的证书路径。这个证书用于向 etcd 服务器证明客户端的身份。

--key：这个参数指定了 etcdctl 客户端的私钥路径。这个私钥用于向 etcd 服务器进行身份验证和加密通信。

--endpoints：这个参数指定了 etcd 集群的地址和端口。在这个例子中，你正在连接到 IP 地址为 192.168.128.11，端口为 2379 的 etcd 服务器。

member list：这是 etcdctl 的子命令，用于列出 etcd 集群的所有成员。

3、获取节点运行状态

```
sh-5.0# etcdctl \
--write-out=table \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 endpoint health
```

```
sh-5.0# etcdctl \
> --write-out=table \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.1.11:2379 endpoint health
+---------------------------+--------+------------+-------+
|         ENDPOINT          | HEALTH |    TOOK    | ERROR |
+---------------------------+--------+------------+-------+
| https://192.168.1.11:2379 |   true | 7.125558ms |       |
+---------------------------+--------+------------+-------+
```

## 3.1.2、Etcd 数据增删查

我们在学习一个数据库系统时，通常需要先学习数据库的“增删改查”操作。那么大家可能会发现，我们小节标题不存在“改”，这是为什么呢？

那是因为我们的 put 操作是新增，如果要新增的 key 或 value 存在时，就会覆盖，就等同于我们的“改”操作。

- 应用示例

**版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。**

示例 1：查看 etcd 中所有的 key

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
get / --prefix --keys-only
```

```
sh-5.0# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> get / --prefix --keys-only
/registry/apiextensions.k8s.io/customresourcedefinitions/bgpconfigurations.crd.projectcalico.org

/registry/apiextensions.k8s.io/customresourcedefinitions/bgppeers.crd.projectcalico.org

/registry/apiextensions.k8s.io/customresourcedefinitions/blockaffinities.crd.projectcalico.org

/registry/apiextensions.k8s.io/customresourcedefinitions/caliconodestatuses.crd.projectcalico.org

/registry/apiextensions.k8s.io/customresourcedefinitions/clusterinformations.crd.projectcalico.org

/registry/apiextensions.k8s.io/customresourcedefinitions/felixconfigurations.crd.projectcalico.org

/registry/apiextensions.k8s.io/customresourcedefinitions/globalnetworkpolicies.crd.projectcalico.org
```

对上面命令的参数说明如下：

get / --prefix --keys-only：这是 etcdctl 的子命令，用于获取 etcd 中的键。--prefix 参数表示只获取以给定前缀开始的键，--keys-only 参数表示只返回键，而不返回值。

示例 2：新增 key:value（etcd 的 put 操作）

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 put foo bar
```

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 put foo bar
OK
sh-5.1#
```

示例 3：查询 key 对应的值（etcd get 操作）

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 get foo
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 get foo
foo
bar
sh-5.1#
```

示例 4：修改 key:value（etcd 的 put 操作）

1、修改 foo 的值为 big

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 put foo big
```

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 put foo big
OK
sh-5.1#
```

2、查询 foo 的值

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
--endpoints=https://192.168.128.11:2379 get foo
```

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 get foo
foo
big
sh-5.1#
```

示例 5：删除 key:value（etcd 的 del 操作）

1、删除 key 为 foo

```
sh-5.0# etcdctl \
--cacert=/etc/kubernetes/pki/etcd/ca.crt \
--cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
--key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
    --endpoints=https://192.168.128.11:2379 del foo
```

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 del foo
1
sh-5.1#
```

2、查询是否删除

```
    sh-5.0# etcdctl \
    --cacert=/etc/kubernetes/pki/etcd/ca.crt \
    --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
    --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
    --endpoints=https://192.168.128.11:2379 get foo
```

```
sh-5.1# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 get foo
sh-5.1#
```

## 3.1、Etcd pod 外的基本操作

在容器内操作 etcd 有一个弊端，很多命令不存在，例如：ls、cat、mv 等这些命令不存在，让我们操作很不舒服。对于非 kubeadm 部署的 k8s 集群，我们也是进行如下操作：

（1）上传 etcd 二进制包，移动 etcdctl 命令到系统/usr/local/bin 下

```
1、上传软件包
    [root@k8s-master01 ~]# ll etcd-v3.4.18-linux-amd64.tar.gz
    -rw-r--r-- 1 root root 17414708 11 月  14 2021 etcd-v3.4.18-linux-amd64.tar.gz

2、解压缩
    [root@k8s-master01 ~]# tar xf etcd-v3.4.18-linux-amd64.tar.gz

3、移动 etcdctl 命令到/usr/local/bin 下
    [root@k8s-master01 ~]# mv etcd-v3.4.18-linux-amd64/etcdctl /usr/local/bin/
```

2、获取 etcd 的成员列表

```
    [root@k8s-master01 ~]# etcdctl \
    --cacert=/etc/kubernetes/pki/etcd/ca.crt \
    --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
    --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
```

```
    --endpoints=https://192.168.128.11:2379 -w table member list
```

```
[root@k8s-master01 ~]# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 -w table member list
+------------------+---------+-------------+---------------------------+---------------------------+------------+
|        ID        | STATUS  |     NAME    |         PEER ADDRS        |        CLIENT ADDRS       | IS LEARNER |
+------------------+---------+-------------+---------------------------+---------------------------+------------+
| efc91ac160b34f04 | started | k8s-master01 | https://192.168.128.11:2380 | https://192.168.128.11:2379 |      false |
+------------------+---------+-------------+---------------------------+---------------------------+------------+
[root@k8s-master01 ~]#
```

## 4、Etcd 备份与恢复

### 4.1、Etcd 快照备份

Etcd 快照备份是一种备份 Etcd 集群数据的方法。Etcd 是一个用于存储 Kubernetes 集群状态的数据库，因此备份 Etcd 集群数据对于在灾难场景下恢复 Kubernetes 集群非常重要。

Etcd 快照备份是通过创建一个快照来完成的，快照包含了所有 Kubernetes 状态和关键信息。快照文件包含所有 Kubernetes 状态和关键信息。为了确保敏感的 Kubernetes 数据的安全，可以对快照文件进行加密。

创建 Etcd 快照的步骤如下：

1、使用 etcdctl 命令创建快照：etcdctl snapshot save <snapshot_file>。这个命令将从使用 etcdctl snapshot save 命令的活动成员中获取快照。

2、可选：如果需要加密快照文件，可以使用如下命令：etcdctl snapshot save --encrypt <encrypted_snapshot_file>。这个命令会创建一个加密的快照文件。

- 应用示例

  创建备份文件存储目录

```
1、创建备份文件存储目录
    [root@k8s-master01 ~]# mkdir /etcd_backup
```

示例 1：为 etcd 创建非加密快照文件

```
1、创建快照文件，文件名为 etcd.data
    [root@k8s-master01 ~]# etcdctl \
    --cacert=/etc/kubernetes/pki/etcd/ca.crt \
    --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
    --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
    --endpoints=https://192.168.128.11:2379 snapshot save /etcd_backup/etcd.data
```

```
[root@k8s-master01 ~]# etcdctl \
> --cacert=/etc/kubernetes/pki/etcd/ca.crt \
> --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
> --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
> --endpoints=https://192.168.128.11:2379 snapshot save /etcd_backup/etcd.data
{"level":"info","ts":1693811723.8379152,"caller":"snapshot/v3_snapshot.go:119","msg":"created temporary db file","path":"/etcd_backup/etcd.data.part"}
{"level":"info","ts":"2023-09-04T03:15:23.843-0400","caller":"clientv3/maintenance.go:200","msg":"opened snapshot stream; downloading"}
{"level":"info","ts":1693811723.843576,"caller":"snapshot/v3_snapshot.go:127","msg":"fetching snapshot","endpoint":"https://192.168.128.11:2379"}
{"level":"info","ts":"2023-09-04T03:15:23.911-0400","caller":"clientv3/maintenance.go:208","msg":"completed snapshot read; closing"}
{"level":"info","ts":1693811723.92192,"caller":"snapshot/v3_snapshot.go:142","msg":"fetched snapshot","endpoint":"https://192.168.128.11:2379","size":"9.7 MB","took":0.083923959}
{"level":"info","ts":1693811723.9219897,"caller":"snapshot/v3_snapshot.go:152","msg":"saved","path":"/etcd_backup/etcd.data"}
Snapshot saved at /etcd_backup/etcd.data
[root@k8s-master01 ~]#
```

2、查看备份文件

```
[root@k8s-master01 ~]# ll /etcd_backup/
total 9500
-rw------- 1 root root 9723936 Sep  4 03:15 etcd.data
```

## 4.2、Etcd 快照还原

Etcd 快照还原是一种从 Etcd 快照中恢复数据的方法。Etcd 是一个用于存储 Kubernetes 集群状态的分布式键值数据库，如果发生故障，可以使用快照来恢复数据。

Etcd 快照还原的步骤如下：

1、确保有可用的快照文件。在执行还原操作之前，需要先获取到 Etcd 的快照文件。

2、停止 Etcd 服务：使用如下命令停止 Etcd 服务：systemctl stop etcd。

3、删除 Etcd 数据目录：在还原操作之前，需要先删除 Etcd 的数据目录，以防止有旧的快照文件存在。使用如下命令删除数据目录：rm -rf /var/lib/etcd/。

4、恢复 etcd 数据目录：etcdctl snapshot restore /backup/snapshot.db --data-dir=/var/lib/etcd

5、启动 Etcd 服务：使用如下命令启动 Etcd 服务：systemctl start etcd。

6、恢复 Etcd 服务数据。

● etcd 恢复演练

（1）创建测试 pod

```
[root@k8s-master01 ~]# kubectl apply -f tomcat.yaml
pod/demo-pod created
service/tomcat created

[root@k8s-master01 ~]# kubectl get pods
NAME       READY   STATUS    RESTARTS   AGE
demo-pod   1/1     Running   0          27s
```

（2）备份 etcd 数据

```
[root@k8s-master01 ~]# mkdir /etcd_backup
[root@k8s-master01 ~]# cd /etcd_backup/

[root@k8s-master01 etcd_backup]# etcdctl \
```

```
    --cacert=/etc/kubernetes/pki/etcd/ca.crt \
    --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
    --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
    --endpoints=https://192.168.128.11:2379 snapshot save /etcd_backup/etcd.db
```

（3）模拟业务 pod 被删除

```
    [root@k8s-master01 etcd_backup]# kubectl delete pods demo-pod
    pod "demo-pod" deleted
    [root@k8s-master01 etcd_backup]# kubectl get pods
    No resources found in default namespace.
```

（4）还原数据

```
1、停止 etcd 数据库
    [root@k8s-master01 etcd_backup]# mv /etc/kubernetes/manifests/etcd.yaml .
    # 检查
    [root@k8s-master01 etcd_backup]# netstat -tlunp | grep 2379

2、移动 etcd 数据目录
    [root@k8s-master01 etcd_backup]# mv /var/lib/etcd /opt/

3、还原 etcd 数据
    [root@k8s-master01 etcd_backup]# etcdctl \
    --cacert=/etc/kubernetes/pki/etcd/ca.crt \
    --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt \
    --key=/etc/kubernetes/pki/etcd/healthcheck-client.key \
    --endpoints=https://192.168.128.11:2379          snapshot          restore
/etcd_backup/etcd.db --data-dir=/var/lib/etcd

4、恢复 etcd
    [root@k8s-master01 etcd_backup]# mv etcd.yaml /etc/kubernetes/manifests/
```

（5）检查业务 pod 是否恢复成功

```
    [root@k8s-master01 etcd_backup]# kubectl get pods
    NAME       READY   STATUS    RESTARTS   AGE
    demo-pod   1/1     Running   0          10m
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。