4.6 实战案例——ZooKeeper 集群部署

4.6.1 案例目标

（1）了解 ZooKeeper 分布式应用程序协调服务。

（2）使用 3 台机器搭建 ZooKeeper 集群。

（3）使用 ZooKeeper 集群。

4.6.2 案例分析

1. 规划节点

ZooKeeper 集群系统的节点规划，见表 4-6-1。

表 4-6-1 节点规划

IP	主机名	节点
172.16.51.23	zookeeper1	集群节点
172.16.51.32	zookeeper2	集群节点
172.16.51.41	zookeeper3	集群节点

2. 基础准备

登录 OpenStack 平台，使用提供的 CentOS_7.2_x86_64_XD.qcow2 镜像，flavor 使用 2vCPU/4GB 内存/50GB 硬盘创建云主机。使用提供的 zookeeper-3.4.14.tar.gz 包和 gpmall-repo 文件夹，安装 ZooKeeper 服务。

4.6.3 案例实施

1. 基础环境配置

（1）主机名配置

使用 secureCRT 对 3 台云主机进行连接。

3 个节点修改主机名为 zookeeper1、zookeeper2、zookeeper3，命令如下：

zookeeper1 节点：

```
[root@localhost ~]# hostnamectl set-hostname zookeeper1
```

zookeeper2 节点：

```
[root@localhost ~]# hostnamectl set-hostname zookeeper2
```

zookeeper3 节点：

```
[root@localhost ~]# hostnamectl set-hostname zookeeper3
```

修改完之后重新连接 secureCRT，并查看主机名：

zookeeper1 节点：

```
[root@zookeeper1 ~]# hostnamectl
   Static hostname: zookeeper1
         Icon name: computer-vm
           Chassis: vm
        Machine ID: dae72fe0cc064eb0b7797f25bfaf69df
           Boot ID: c642ea4be7d349d0a929e557f23ce3dc
    Virtualization: kvm
  Operating System: CentOS Linux 7 (Core)
       CPE OS Name: cpe:/o:centos:centos:7
            Kernel: Linux 3.10.0-229.el7.x86_64
      Architecture: x86_64
```

zookeeper2 节点：

```
[root@zookeeper2 ~]# hostnamectl
   Static hostname: zookeeper2
         Icon name: computer-vm
           Chassis: vm
        Machine ID: dae72fe0cc064eb0b7797f25bfaf69df
           Boot ID: cfcaf92af7a44028a098dc4792b441f4
    Virtualization: kvm
  Operating System: CentOS Linux 7 (Core)
       CPE OS Name: cpe:/o:centos:centos:7
            Kernel: Linux 3.10.0-229.el7.x86_64
      Architecture: x86_64
```

zookeeper3 节点：

```
[root@zookeeper3 ~]# hostnamectl
   Static hostname: zookeeper3
```

```
                Icon name: computer-vm
                  Chassis: vm
               Machine ID: dae72fe0cc064eb0b7797f25bfaf69df
                  Boot ID: cff5bbd45243451e88d14e1ec75098c0
      Virtualization: kvm
    Operating System: CentOS Linux 7 (Core)
         CPE OS Name: cpe:/o:centos:centos:7
              Kernel: Linux 3.10.0-229.el7.x86_64
        Architecture: x86_64
```

（2）配置 hosts 文件

3 个节点修改/etc/hosts 文件，3 个节点均修改成如下代码所示：

```
# vi /etc/hosts
172.16.51.23 zookeeper1
172.16.51.32 zookeeper2
172.16.51.41 zookeeper3
```

（3）配置 YUM 源

将提供的 gpmall-repo 目录上传至 3 个节点的/opt 目录下，首先将 3 个节点/etc/yum.repo.d 目录下的文件移动到/media 目录下，命令如下：

```
# mv /etc/yum.repos.d/* /media/
```

在 3 个节点上创建/etc/yum.repo.d/local.repo，文件内容如下：

```
# cat /etc/yum.repos.d/local.repo
[gpmall]
name=gpmall
baseurl=file:///opt/gpmall-repo
gpgcheck=0
enabled=1
# yum clean all
# yum list
```

2. 搭建 ZooKeeper 集群

（1）安装 JDK 环境

3 个节点安装 Java JDK 环境，3 个节点均执行命令如下：

```
# yum install -y java-1.8.0-openjdk java-1.8.0-openjdk-devel
# java -version
openjdk version "1.8.0_222"
OpenJDK Runtime Environment (build 1.8.0_222-b10)
OpenJDK 64-Bit Server VM (build 25.222-b10, mixed mode)
```

（2）解压 ZooKeeper 软件包

将 zookeeper-3.4.14.tar.gz 软件包上传至 3 个节点的/root 目录下，进行解压操作，3 个节点均执行命令如下：

```
# tar -zxvf zookeeper-3.4.14.tar.gz
```

（3）修改 3 个节点配置文件

在 zookeeper1 节点，进入 zookeeper-3.4.14/conf 目录下，修改 zoo_sample.cfg 文件为 zoo.cfg，并编辑该文件内容如下：

```
[root@zookeeper1 conf]# vi zoo.cfg
[root@zookeeper1 conf]# grep -n '^'[a-Z] zoo.cfg
2:tickTime=2000
5:initLimit=10
8:syncLimit=5
12:dataDir=/tmp/zookeeper
14:clientPort=2181
29:server.1=172.16.51.23:2888:3888
30:server.2=172.16.51.32:2888:3888
31:server.3=172.16.51.41:2888:3888
```

命令解析：

● initLimit：ZooKeeper 集群模式下包含多个 zk 进程，其中一个进程为 leader，余下的进程为 follower。当 follower 最初与 leader 建立连接时，它们之间会传输相当多的数据，尤其是 follower 的数据落后 leader 很多。initLimit 配置 follower 与 leader 之间建立连接后进行同步的最长时间。

● syncLimit：配置 follower 和 leader 之间发送消息，请求和应答的最大时间长度。

● tickTime：tickTime 则是上述两个超时配置的基本单位，例如对于 initLimit，其配置值为 5，说明其超时时间为 2000ms * 5 = 10 秒。

● server.id=host:port1:port2：其中 id 为一个数字，表示 zk 进程的 id，这个 id 也是 dataDir 目录下 myid 文件的内容。host 是该 zk 进程所在的 IP 地址，port1 表示 follower 和 leader 交换消息所使用的端口，port2 表示选举 leader 所使用的端口。

● dataDir：其配置的含义跟单机模式下的含义类似，不同的是集群模式下还有一个 myid 文件。myid 文件的内容只有一行，且内容只能为 1 - 255 之间的数字，这个数字亦即上面介绍 server.id 中的 id，表示 zk 进程的 id。

注意：zookeeper2 和 zookeeper3 节点的操作与修改的配置和 zookeeper1 节点一样。

（4）创建 myid 文件

在 3 台机器 dataDir 目录（此处为/tmp/zookeeper）下，分别创建一个 myid 文件，文件内容分别只有一行，其内容为 1，2，3。即文件中只有一个数字，这个数字即为上面 zoo.cfg 配置文件中指定的值。ZooKeeper 是根据该文件来决定 ZooKeeper 集群各个机器的身份分配。

创建 myid 文件，命令如下：

zookeeper1 节点：

```
[root@zookeeper1 ~]# mkdir /tmp/zookeeper
[root@zookeeper1 ~]# vi /tmp/zookeeper/myid
[root@zookeeper1 ~]# cat /tmp/zookeeper/myid
1
```

zookeeper2 节点：

```
[root@zookeeper2 ~]# mkdir /tmp/zookeeper
[root@zookeeper2 ~]# vi /tmp/zookeeper/myid
[root@zookeeper2 ~]# cat /tmp/zookeeper/myid
2
```

zookeeper3 节点：

```
[root@zookeeper3 ~]# mkdir /tmp/zookeeper
[root@zookeeper3 ~]# vi /tmp/zookeeper/myid
[root@zookeeper3 ~]# cat /tmp/zookeeper/myid
3
```

（5）启动 ZooKeeper 服务

在 3 台机器的 zookeeper-3.4.14/bin 目录下执行命令如下：

zookeeper1 节点：

```
[root@zookeeper1 bin]# ./zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /root/zookeeper-3.4.14/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[root@zookeeper1 bin]# ./zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /root/zookeeper-3.4.14/bin/../conf/zoo.cfg
Mode: follower
```

zookeeper2 节点：

```
[root@zookeeper2 bin]# ./zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /root/zookeeper-3.4.14/bin/../conf/zoo.cfg
Starting zookeeper ... already running as process 10175.
[root@zookeeper2 bin]# ./zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /root/zookeeper-3.4.14/bin/../conf/zoo.cfg
Mode: leader
```

zookeeper3 节点：

```
[root@zookeeper3 bin]# ./zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /root/zookeeper-3.4.14/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[root@zookeeper3 bin]# ./zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /root/zookeeper-3.4.14/bin/../conf/zoo.cfg
Mode: follower
```

可以看到，3 个节点，zookeeper2 为 leader，其他的都是 follower。

至此，ZooKeeper 集群配置完毕。

注意：查看状态出现问题时，所有节点都启动一下，再查看状态。