分布式协调服务系统 ZooKeeper

一、实训说明：

本次实训，主要是搭建分布式协调服务系统 zookeeper。zookeeper 它是一个为分布式应用提供一致性服务的软件，提供的功能包括：配置维护、域名服务、分布式同步、组服务等。

二、实训环境

1）已经配置完成的 Hadoop 完全分布式环境
2）使用软件：
Zookeeper： zookeeper-3.4.13.tar.gz
下载地址（版本更新路径会导致地址不存在）：
http://archive.apache.org/dist/zookeeper/zookeeper-3.4.13/zookeeper-3.4.13.tar.gz

三、实训内容

以下操作均在 hadoop 用户下进行

1. zookeeper 安装

1）解压安装包（master 节点）

```
[hadoop@master ~]$ sudo tar -zxvf /home/package/zookeeper-3.4.13.ta-r.gz -C /usr
```

2）重命名安装路径（master 节点）

```
[hadoop@master ~]$ sudo mv /usr/zookeeper-3.4.13/ /usr/zookeeper
```

```
[hadoop@master ~]$ sudo mv /usr/zookeeper-3.4.13/ /usr/zookeeper
[hadoop@master ~]$ 
```

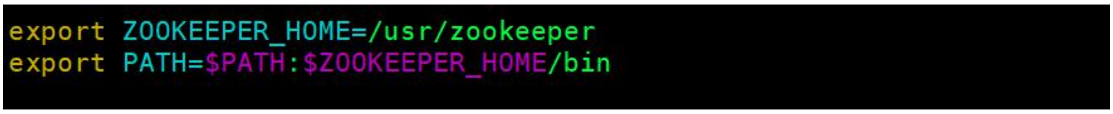
3）添加环境变量（所有节点）

```
[hadoop@master ~]$ sudo vim /etc/profile
```

添加以下环境变量

```
export ZOOKEEPER_HOME=/usr/zookeeper

export PATH=$PATH:$ZOOKEEPER_HOME/bin
```



```
export ZOOKEEPER_HOME=/usr/zookeeper
export PATH=$PATH:$ZOOKEEPER_HOME/bin
```

4）使环境变量生效（所有节点）

```
[hadoop@master ~]$ source  /etc/profile
```

5）修改 zoo.cfg 配置文件（master 节点）
因为配置不存在，我们先从模板文件复制一份

```
[hadoop@master ~]$ sudo cp $ZOOKEEPER_HOME/conf/zoo_sample.cfg
$ZOOKEEPER_HOME/conf/zoo.cfg
```

修改配置文件：

```
[hadoop@master ~]$ sudo vim $ZOOKEEPER_HOME/conf/zoo.cfg
```

修改或添加以下配置信息：

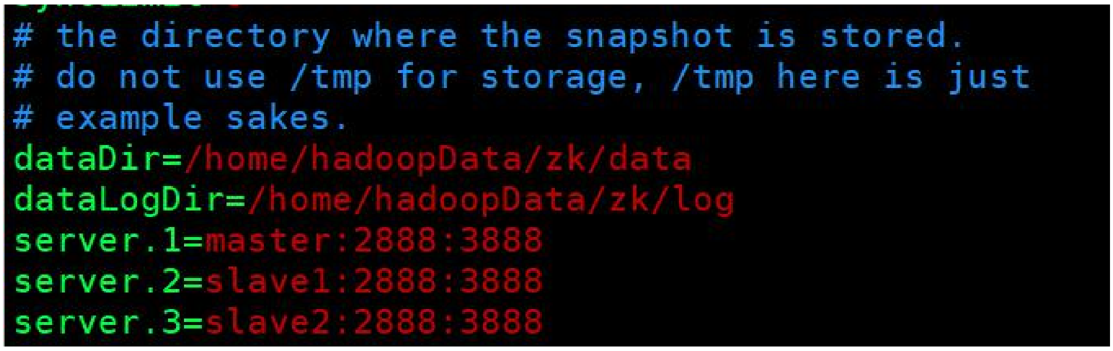
```
dataDir=/home/hadoopData/zk/data

dataLogDir=/home/hadoopData/zk/log

server.1=master:2888:3888

server.2=slave1:2888:3888

server.3=slave2:2888:3888
```



```
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
dataDir=/home/hadoopData/zk/data
dataLogDir=/home/hadoopData/zk/log
server.1=master:2888:3888
server.2=slave1:2888:3888
server.3=slave2:2888:3888
```

6）将配置文件分发到 slave1 和 slave2

```
[hadoop@master ~]$ sudo scp -r  /usr/zookeeper/ slave1:/usr/

[hadoop@master ~]$ sudo scp -r  /usr/zookeeper/ slave2:/usr/
```

7）创建 dataDir 和 dataLogDir（所有节点）

```
[hadoop@master ~]$ mkdir -p /home/hadoopData/zk/data

[hadoop@master ~]$ mkdir -p /home/hadoopData/zk/log
```

8）创建 myid 文件（所有节点）
在 dataDir 目录下创建 myid 文件，该文件的内容根据 server 定义的不同而不同，如 server.1（master 节点）该文件的内容是 1，server.2（slave1 节点）该文件内容是 2，server.3（slave2 节点）该文件内容是 3。
master 节点：

```
[hadoop@master ~]$ echo "1" >> /home/hadoopData/zk/data/myid
```

slave1 节点：

```
[hadoop@slave1 ~]$ echo "2" >> /home/hadoopData/zk/data/myid
```

slave2 节点：

```
[hadoop@slave2 ~]$ echo "3" >> /home/hadoopData/zk/data/myid
```

9）修改 zookeeper 目录权限（所有节点）

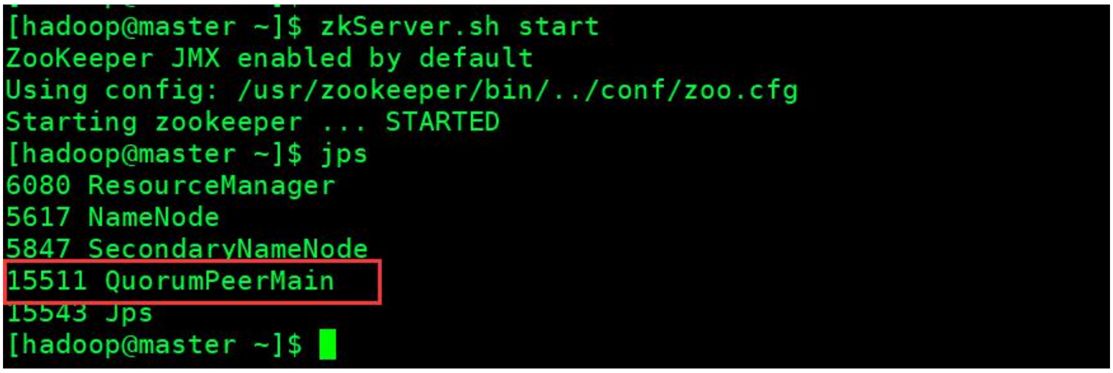
```
[hadoop@master ~]$ sudo chown -R  hadoop:hadoop /usr/zookeeper/
```

10）启动 zookeeper，并查看守护进程

每个节点都要分别启动

```
[hadoop@master ~]$ zkServer.sh start

[hadoop@master ~]$ jps
```



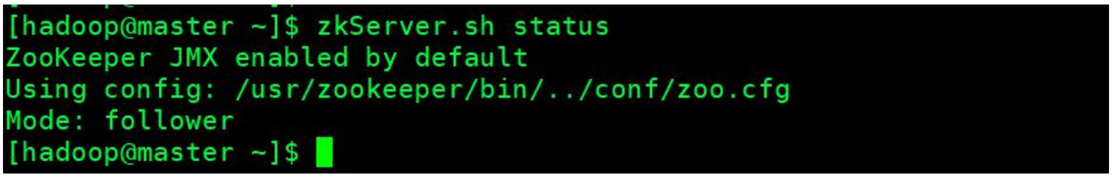
```
[hadoop@master ~]$ zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[hadoop@master ~]$ jps
6080 ResourceManager
5617 NameNode
5847 SecondaryNameNode
15511 QuorumPeerMain
15543 Jps
[hadoop@master ~]$
```

2. 验证测试

1）查看每个节点 zookeeper 状态

master 节点：

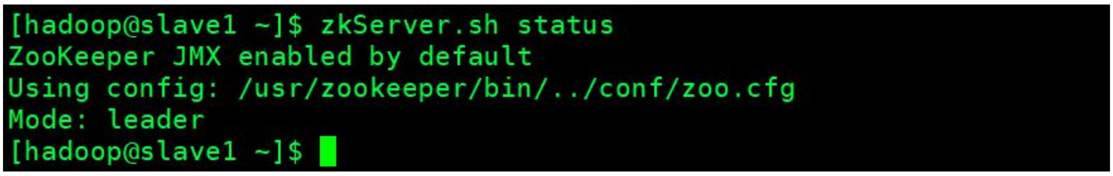
```
[hadoop@master ~]$ zkServer.sh status
```



```
[hadoop@master ~]$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Mode: follower
[hadoop@master ~]$
```

slave1 节点：

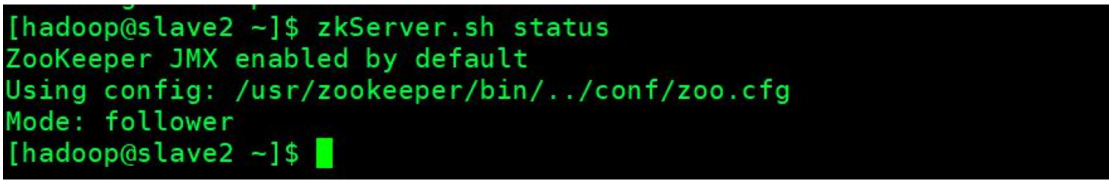
```
[hadoop@slave1 ~]$ zkServer.sh status
```



```
[hadoop@slave1 ~]$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Mode: leader
[hadoop@slave1 ~]$
```

slave2 节点：

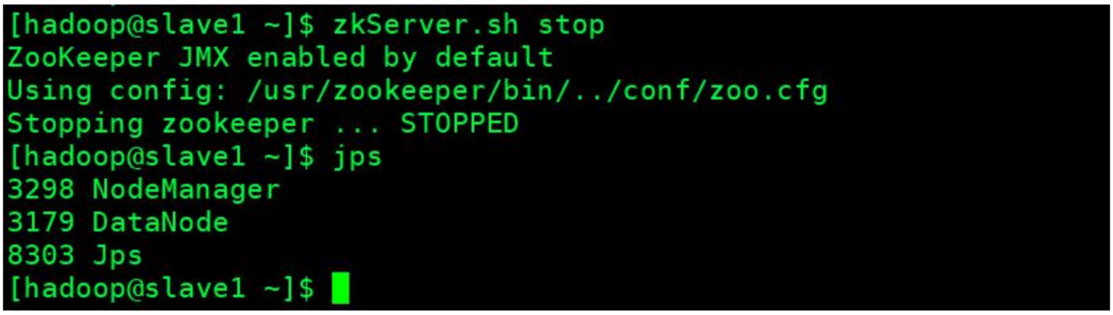
```
[hadoop@slave2 ~]$ zkServer.sh status
```



```
[hadoop@slave2 ~]$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Mode: follower
[hadoop@slave2 ~]$
```

2）关闭 leader 模式节点的 zookeeper 服务

```
[hadoop@slave1 ~]$ zkServer.sh stop
```

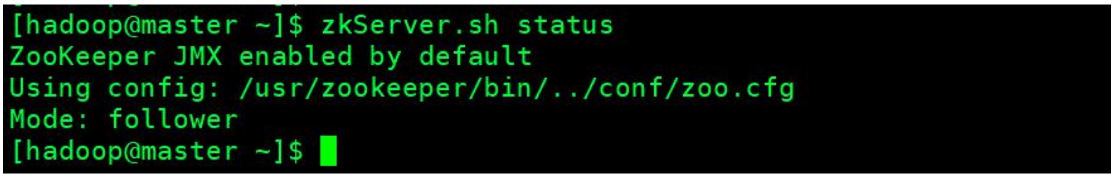


```
[hadoop@slave1 ~]$ zkServer.sh stop
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Stopping zookeeper ... STOPPED
[hadoop@slave1 ~]$ jps
3298 NodeManager
3179 DataNode
8303 Jps
[hadoop@slave1 ~]$
```

3）查看另外两个节点的状态：

slave1：

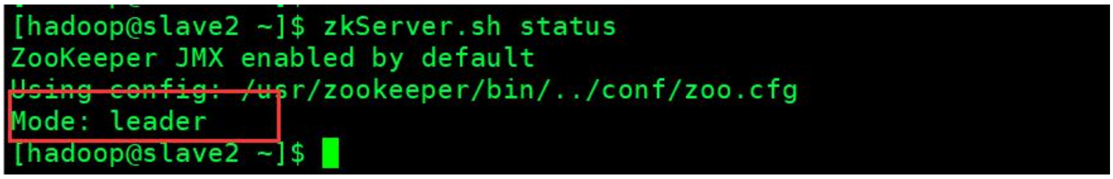
```
[hadoop@master ~]$ zkServer.sh status
```



```
[hadoop@master ~]$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Mode: follower
[hadoop@master ~]$
```

slave2：

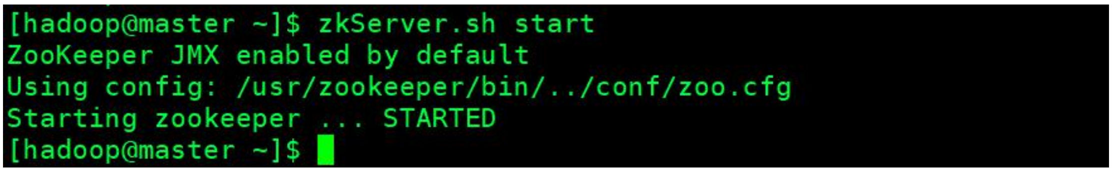
```
[hadoop@slave2 ~]$ zkServer.sh status
```



```
[hadoop@slave2 ~]$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Mode: leader
[hadoop@slave2 ~]$
```

4）开启 slave1 节点的 zookeeper 服务

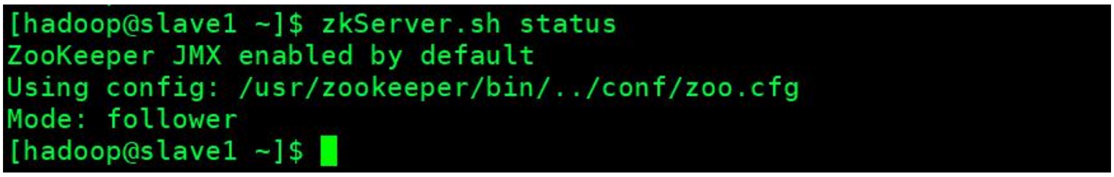
```
[hadoop@master ~]$ zkServer.sh start
```



```
[hadoop@master ~]$ zkServer.sh start
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Starting zookeeper ... STARTED
[hadoop@master ~]$
```

5）查看 slave1 节点状态

```
[hadoop@slave1 ~]$ zkServer.sh status
```



```
[hadoop@slave1 ~]$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/zookeeper/bin/../conf/zoo.cfg
Mode: follower
[hadoop@slave1 ~]$
```

可以看到，当 leader 节点挂掉的时候，zookeeper 会自动从 follower 节点选举出来一个 leader，也证明了我们 zookeeper 服务是已经安装完成了

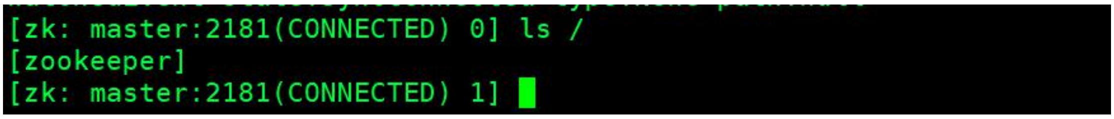
3. Zookeeper shell 命令使用

1）打开客户端连接

```
[hadoop@master ~]$ zkCli.sh -server master:2181
```

2）查看节点：ls

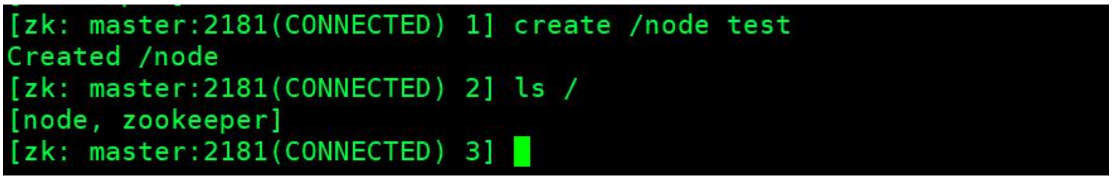
```
[zk: master:2181(CONNECTED) 0] ls /
```



```
[zk: master:2181(CONNECTED) 0] ls /
[zookeeper]
[zk: master:2181(CONNECTED) 1] 
```

3）创建一个 znode 节点：create

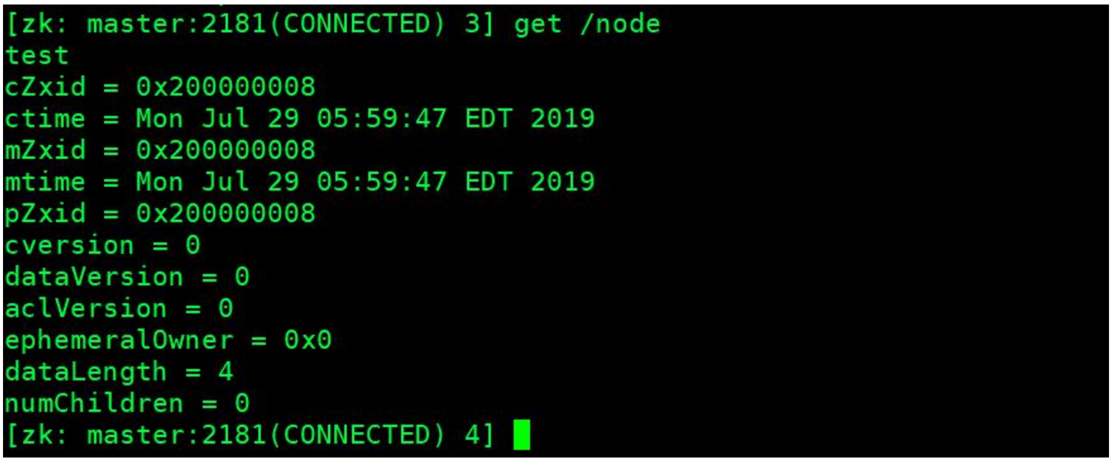
```
[zk: master:2181(CONNECTED) 1] create /node test
```



```
[zk: master:2181(CONNECTED) 1] create /node test
Created /node
[zk: master:2181(CONNECTED) 2] ls /
[node, zookeeper]
[zk: master:2181(CONNECTED) 3] 
```

4）获取节点信息：get

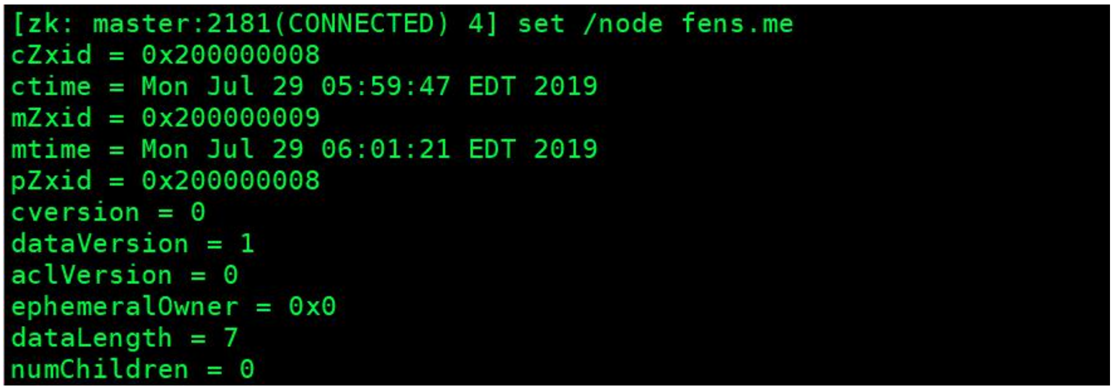
```
[zk: master:2181(CONNECTED) 3] get /node
```



```
[zk: master:2181(CONNECTED) 3] get /node
test
cZxid = 0x200000008
ctime = Mon Jul 29 05:59:47 EDT 2019
mZxid = 0x200000008
mtime = Mon Jul 29 05:59:47 EDT 2019
pZxid = 0x200000008
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 4
numChildren = 0
[zk: master:2181(CONNECTED) 4] 
```

5）修改节点数据：set

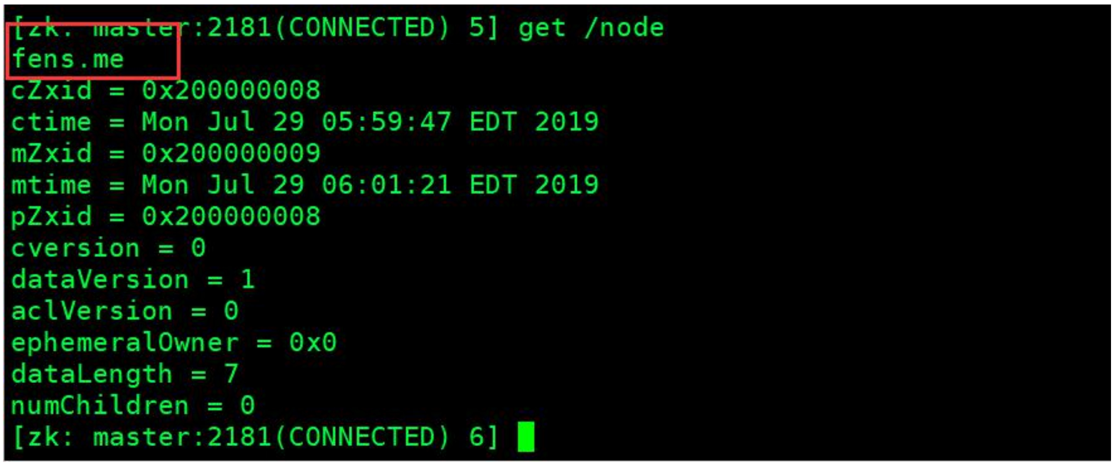
```
[zk: master:2181(CONNECTED) 4] set /node fens.me
```



```
[zk: master:2181(CONNECTED) 4] set /node fens.me
cZxid = 0x200000008
ctime = Mon Jul 29 05:59:47 EDT 2019
mZxid = 0x200000009
mtime = Mon Jul 29 06:01:21 EDT 2019
pZxid = 0x200000008
cversion = 0
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 7
numChildren = 0
```

6）再次查看节点信息

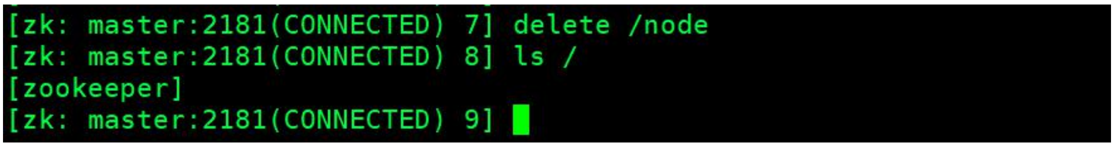
```
[zk: master:2181(CONNECTED) 5] get /node
```



```
[zk: master:2181(CONNECTED) 5] get /node
fens.me
cZxid = 0x200000008
ctime = Mon Jul 29 05:59:47 EDT 2019
mZxid = 0x200000009
mtime = Mon Jul 29 06:01:21 EDT 2019
pZxid = 0x200000008
cversion = 0
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 7
numChildren = 0
[zk: master:2181(CONNECTED) 6]
```

7）删除节点信息

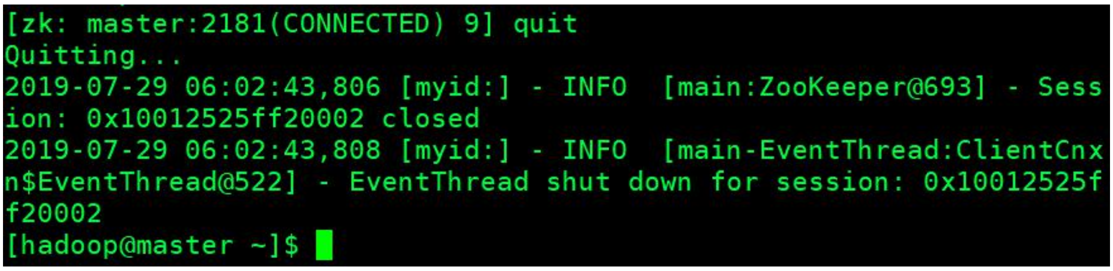
```
[zk: master:2181(CONNECTED) 7] delete /node
```



```
[zk: master:2181(CONNECTED) 7] delete /node
[zk: master:2181(CONNECTED) 8] ls /
[zookeeper]
[zk: master:2181(CONNECTED) 9]
```

8）退出客户端连接

```
[zk: master:2181(CONNECTED) 9] quit
```



```
[zk: master:2181(CONNECTED) 9] quit
Quitting...
2019-07-29 06:02:43,806 [myid:] - INFO  [main:ZooKeeper@693] - Sess
ion: 0x10012525ff20002 closed
2019-07-29 06:02:43,808 [myid:] - INFO  [main-EventThread:ClientCnx
n$EventThread@522] - EventThread shut down for session: 0x10012525f
f20002
[hadoop@master ~]$
```