基于Zookeeper搭建Hadoop高可用集群

> 微信公众号：小康新鲜事儿

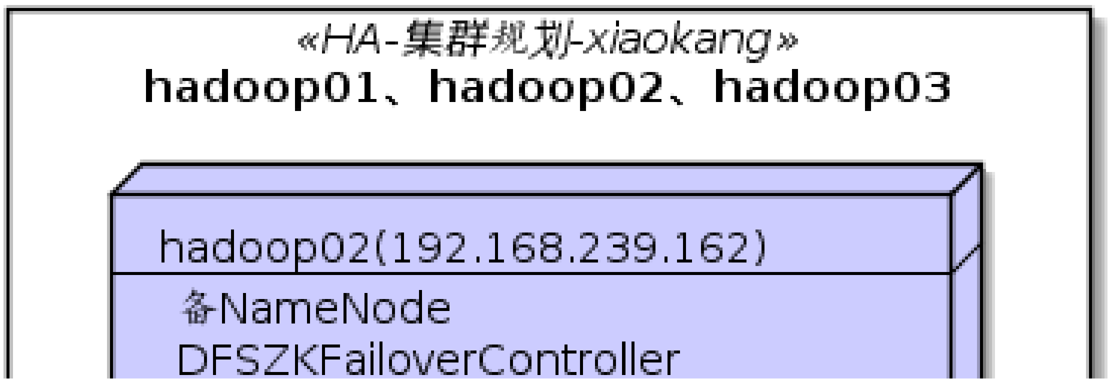
一、前置准备

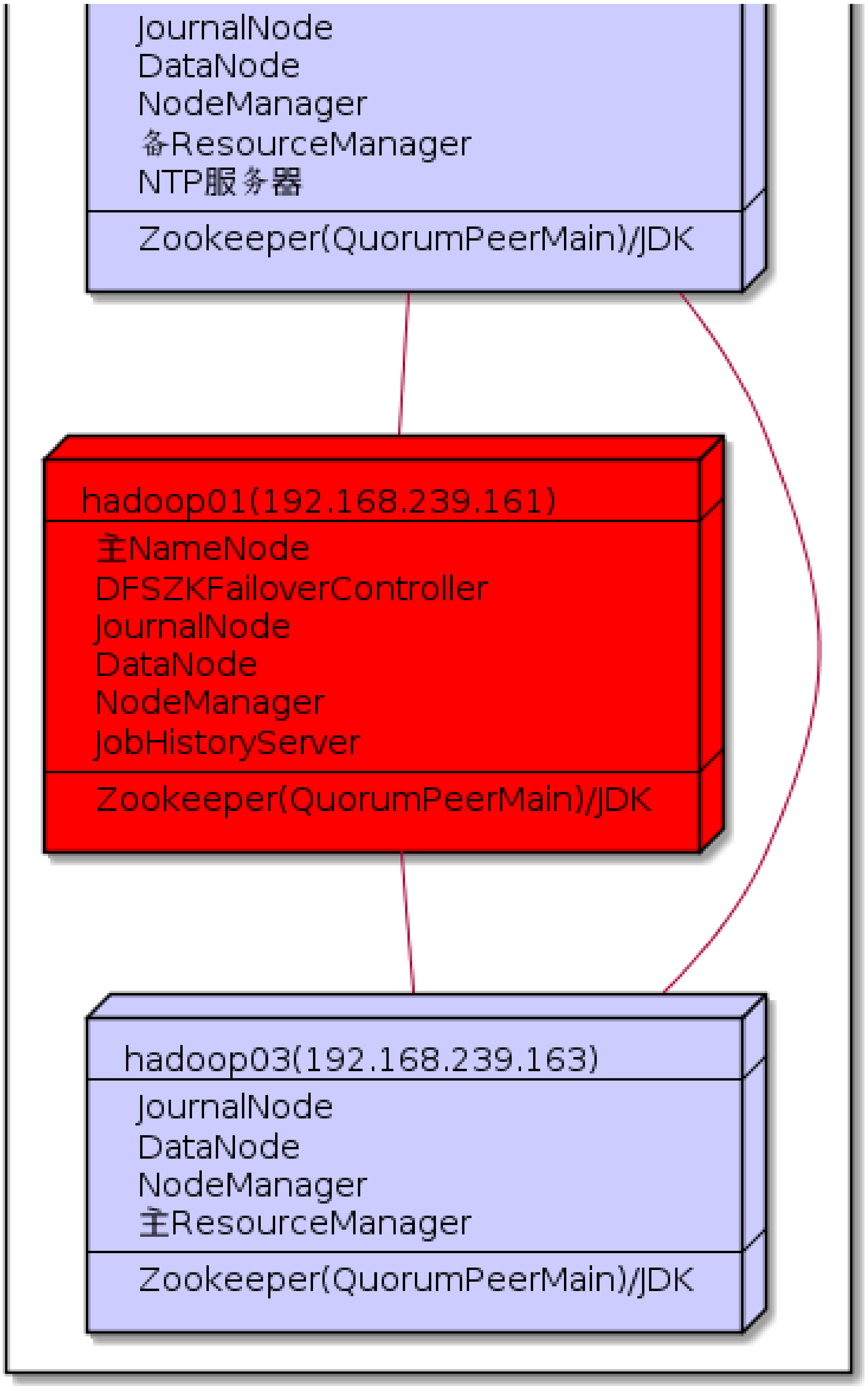
Hadoop前置准备：Hadoop前置准备

Hadoop完全分布式集群环境搭建：Hadoop完全分布式集群环境搭建

Zookeeper集群环境搭建：Zookeeper集群环境搭建

二、集群规划





三、集群配置

先创建好所需目录

```
[xiaokang@hadoop01 ~]$ mkdir -p /opt/software/hadoop-2.7.7/tmp
[xiaokang@hadoop01 ~]$ mkdir -p /opt/software/hadoop-2.7.7/dfs/journalnode_data
[xiaokang@hadoop01 ~]$ mkdir -p /opt/software/hadoop-2.7.7/dfs/edits
[xiaokang@hadoop01 ~]$ mkdir -p /opt/software/hadoop-2.7.7/dfs/datanode_data
[xiaokang@hadoop01 ~]$ mkdir -p /opt/software/hadoop-2.7.7/dfs/namenode_data
```

1. hadoop-env.sh

```
export JAVA_HOME=/opt/moudle/jdk1.8.0_191
export HADOOP_CONF_DIR=/opt/software/hadoop-2.7.7/etc/hadoop
```

2. core-site.xml

```
<configuration>
    <property>
        <!--指定hadoop集群在zookeeper上注册的节点名-->
        <name>fs.defaultFS</name>
        <value>hdfs://hacluster</value>
    </property>
    <property>
        <!--用来指定hadoop运行时产生文件的存放目录-->
        <name>hadoop.tmp.dir</name>
        <value>file:///opt/software/hadoop-2.7.7/tmp</value>
    </property>
    <property>
        <!--设置缓存大小，默认4kb-->
        <name>io.file.buffer.size</name>
        <value>4096</value>
    </property>
    <property>
        <!--指定zookeeper的存放地址 -->
        <name>ha.zookeeper.quorum</name>
        <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
    </property>
</configuration>
```

3. hdfs-site.xml

```
<configuration>
    <property>
        <!--数据块默认大小128M-->
        <name>dfs.block.size</name>
        <value>134217728</value>
    </property>
    <property>
        <!--副本数量，不配置的话默认为3-->
        <name>dfs.replication</name>
        <value>3</value>
    </property>
    <property>
        <!--namenode节点数据（元数据）的存放位置-->
        <name>dfs.name.dir</name>
        <value>file:///opt/software/hadoop-2.7.7/dfs/namenode_data</value>
    </property>
    <property>
        <!--datanode节点数据（元数据）的存放位置-->
        <name>dfs.data.dir</name>
        <value>file:///opt/software/hadoop-2.7.7/dfs/datanode_data</value>
    </property>
    <property>
        <name>dfs.webhdfs.enabled</name>
        <value>true</value>
    </property>
    <property>
        <name>dfs.datanode.max.transfer.threads</name>
        <value>4096</value>
        </property>
    <property>
        <!--指定hadoop集群在zookeeper上注册的节点名-->
        <name>dfs.nameservices</name>
        <value>hacluster</value>
    </property>
    <property>
        <!-- hacluster集群下有两个namenode，分别为nn1,nn2 -->
        <name>dfs.ha.namenodes.hacluster</name>
        <value>nn1,nn2</value>
    </property>
    <!-- nn1的rpc、servicepc和http通信 -->
    <property>
        <name>dfs.namenode.rpc-address.hacluster.nn1</name>
        <value>hadoop01:9000</value>
    </property>
    <property>
        <name>dfs.namenode.servicepc-address.hacluster.nn1</name>
        <value>hadoop01:53310</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hacluster.nn1</name>
        <value>hadoop01:50070</value>
    </property>
    <!-- nn2的rpc、servicepc和http通信 -->
    <property>
        <name>dfs.namenode.rpc-address.hacluster.nn2</name>
        <value>hadoop02:9000</value>
    </property>
    <property>
        <name>dfs.namenode.servicepc-address.hacluster.nn2</name>
        <value>hadoop02:53310</value>
    </property>
    <property>
        <name>dfs.namenode.http-address.hacluster.nn2</name>
        <value>hadoop02:50070</value>
    </property>
    <property>
        <!-- 指定namenode的元数据在JournalNode上存放的位置 -->
        <name>dfs.namenode.shared.edits.dir</name>

<value>qjournal://hadoop01:8485;hadoop02:8485;hadoop03:8485/hacluster</value>
    </property>
    <property>
        <!-- 指定JournalNode在本地磁盘存放数据的位置 -->
        <name>dfs.journalnode.edits.dir</name>
        <value>/opt/software/hadoop-2.7.7/dfs/journalnode_data</value>
    </property>
    <property>
        <!-- namenode操作日志的存放位置 -->
        <name>dfs.namenode.edits.dir</name>
        <value>/opt/software/hadoop-2.7.7/dfs/edits</value>
    </property>
    <property>
        <!-- 开启namenode故障转移自动切换 -->
        <name>dfs.ha.automatic-failover.enabled</name>
        <value>true</value>
    </property>
    <property>
        <!-- 配置失败自动切换实现方式 -->
        <name>dfs.client.failover.proxy.provider.hacluster</name>

 <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvide
r</value>
    </property>
    <property>
        <!-- 配置隔离机制 -->
        <name>dfs.ha.fencing.methods</name>
        <value>sshfence</value>
    </property>
    <property>
        <!-- 使用隔离机制需要SSH免密登录 -->
        <name>dfs.ha.fencing.ssh.private-key-files</name>
        <value>/home/xiaokang/.ssh/id_rsa</value>
    </property>
    <property>
        <!--hdfs文件操作权限,false为不验证-->
        <name>dfs.permissions</name>
        <value>false</value>
    </property>
</configuration>
```

4. mapred-site.xml

```
<configuration>
    <property>
        <!--指定mapreduce运行在yarn上-->
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <!--配置任务历史服务器地址-->
        <name>mapreduce.jobhistory.address</name>
        <value>hadoop01:10020</value>
    </property>
    <property>
        <!--配置任务历史服务器web-UI地址-->
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>hadoop01:19888</value>
    </property>
    <property>
        <!--开启uber模式-->
        <name>mapreduce.job.ubertask.enable</name>
        <value>true</value>
    </property>
</configuration>
```

5. yarn-site.xml

```
<configuration>
    <property>
        <!-- 开启Yarn高可用 -->
        <name>yarn.resourcemanager.ha.enabled</name>
        <value>true</value>
    </property>
    <property>
        <!-- 指定Yarn集群在zookeeper上注册的节点名 -->
        <name>yarn.resourcemanager.cluster-id</name>
        <value>hayarn</value>
    </property>
    <property>
        <!-- 指定两个ResourceManager的名称 -->
        <name>yarn.resourcemanager.ha.rm-ids</name>
        <value>rm1,rm2</value>
    </property>
    <property>
        <!-- 指定rm1的主机 -->
        <name>yarn.resourcemanager.hostname.rm1</name>
        <value>hadoop02</value>
    </property>
    <property>
        <!-- 指定rm2的主机 -->
        <name>yarn.resourcemanager.hostname.rm2</name>
        <value>hadoop03</value>
    </property>
    <property>
        <!-- 配置zookeeper的地址 -->
        <name>yarn.resourcemanager.zk-address</name>
        <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
    </property>
    <property>
        <!-- 开启Yarn恢复机制 -->
        <name>yarn.resourcemanager.recovery.enabled</name>
        <value>true</value>
    </property>
    <property>
        <!-- 配置执行ResourceManager恢复机制实现类 -->
        <name>yarn.resourcemanager.store.class</name>

 <value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</v
alue>
    </property>
    <property>
        <!--指定主resourcemanager的地址-->
        <name>yarn.resourcemanager.hostname</name>
        <value>hadoop03</value>
    </property>
    <property>
        <!--NodeManager获取数据的方式-->
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <!--开启日志聚集功能-->
        <name>yarn.log-aggregation-enable</name>
        <value>true</value>
    </property>
    <property>
        <!--配置日志保留7天-->
        <name>yarn.log-aggregation.retain-seconds</name>
        <value>604800</value>
    </property>
</configuration>
```

6. slaves

```
hadoop01
hadoop02
hadoop03
```

将 Hadoop 安装包分发到其他两台服务器，分发后建议在这两台服务器上也配置一下 Hadoop 的环境变量。

```
# 将安装包分发到hadoop02
[xiaokang@hadoop01 ~]$ scp -r /opt/software/hadoop-2.7.7/
xiaokang@hadoop02:/opt/software/
# 将安装包分发到hadoop03
[xiaokang@hadoop01 hadoop]$ scp -r /opt/software/hadoop-2.7.7/
xiaokang@hadoop03:/opt/software/
```

四、启动集群（初始化工作）

1. 启动3个Zookeeper

```
[xiaokang@hadoop01 ~]$ zkServer.sh start
[xiaokang@hadoop02 ~]$ zkServer.sh start
[xiaokang@hadoop03 ~]$ zkServer.sh start
```

2. 启动3个JournalNode

```
[xiaokang@hadoop01 ~]$ hadoop-daemon.sh start journalnode
[xiaokang@hadoop02 ~]$ hadoop-daemon.sh start journalnode
[xiaokang@hadoop03 ~]$ hadoop-daemon.sh start journalnode
```

3. 格式化NameNode

```
【仅hadoop01】
[xiaokang@hadoop01 ~]$ hdfs namenode -format
```

4. 复制hadoop01上的NameNode的元数据到hadoop02

```
[xiaokang@hadoop01 ~]$ scp -r /opt/software/hadoop-2.7.7/dfs/namenode_data/current/ xiaokang@hadoop02:/opt/software/hadoop-2.7.7/dfs/namenode_data/
```

5. 在NameNode节点(hadoop01或hadoop02)格式化zkfc

```
【二者选其一即可】
[xiaokang@hadoop01 ~]$ hdfs zkfc -formatZK
或
[xiaokang@hadoop02 ~]$ hdfs zkfc -formatZK
```

6. 在hadoop01上启动HDFS相关服务

```
[xiaokang@hadoop01 ~]$ start-dfs.sh

Starting namenodes on [hadoop01 hadoop02]
hadoop02: starting namenode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-namenode-hadoop02.out
hadoop01: starting namenode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-namenode-hadoop01.out
hadoop03: starting datanode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-datanode-hadoop03.out
hadoop02: starting datanode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-datanode-hadoop02.out
hadoop01: starting datanode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-datanode-hadoop01.out
Starting journal nodes [hadoop01 hadoop02 hadoop03]
hadoop02: journalnode running as process 7546. Stop it first.
hadoop01: journalnode running as process 7827. Stop it first.
hadoop03: journalnode running as process 7781. Stop it first.
Starting ZK Failover Controllers on NN hosts [hadoop01 hadoop02]
hadoop01: starting zkfc, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-zkfc-hadoop01.out
hadoop02: starting zkfc, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-zkfc-hadoop02.out
```

7. 在hadoop03上启动YARN相关服务

```
[xiaokang@hadoop03 ~]$ start-yarn.sh
```

8. 最后单独启动hadoop01的历史任务服务器和hadoop02的ResourceManager

```
[xiaokang@hadoop01 ~]$ mr-jobhistory-daemon.sh start historyserver
[xiaokang@hadoop02 ~]$ yarn-daemon.sh start resourcemanager
```

五、查看集群

1. jps进程查看

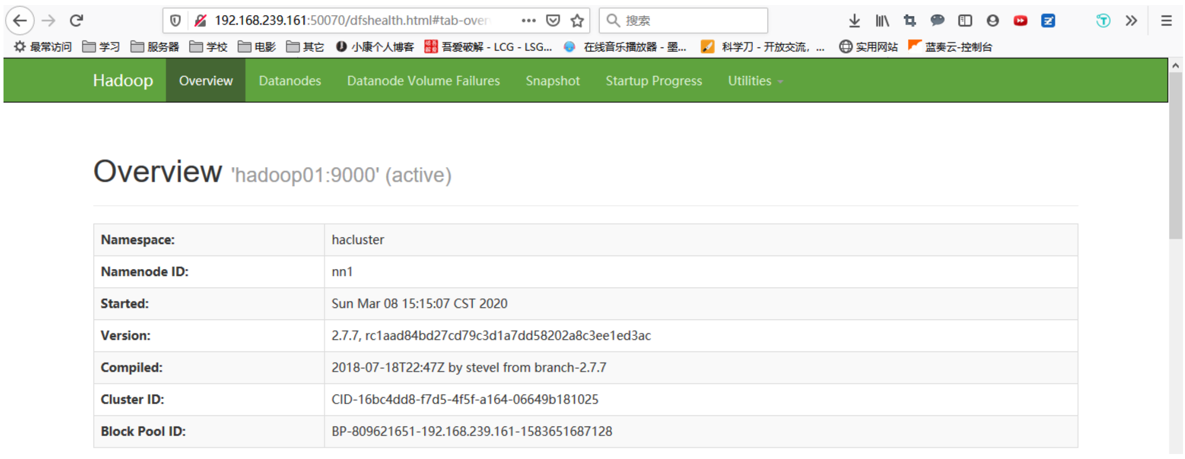
```
[xiaokang@hadoop01 ~]$ jps
```

```
8227 QuorumPeerMain
8916 DataNode
8663 JournalNode
8791 NameNode
9035 DFSZKFailoverController
11048 JobHistoryServer
9147 NodeManager
9260 Jps

[xiaokang@hadoop02 ~]$ jps
7538 QuorumPeerMain
8214 NodeManager
7802 JournalNode
8010 DataNode
8122 DFSZKFailoverController
8346 ResourceManager
8395 Jps
7916 NameNode

[xiaokang@hadoop03 ~]$ jps
8897 Jps
8343 DataNode
8472 ResourceManager
8249 JournalNode
7994 QuorumPeerMain
8575 NodeManager

【查看NameNode的状态】
[xiaokang@hadoop01 ~]$ hdfs haadmin -getServiceState nn1
active
[xiaokang@hadoop01 ~]$ hdfs haadmin -getServiceState nn2
standby
【查看ResourceManager的状态】
[xiaokang@hadoop03 ~]$ yarn rmadmin -getServiceState rm1
standby
[xiaokang@hadoop03 ~]$ yarn rmadmin -getServiceState rm2
active
```

2. WebUI查看

HDFS 和 YARN 的端口号分别为 `50070` 和 `8088`，界面应该如下：

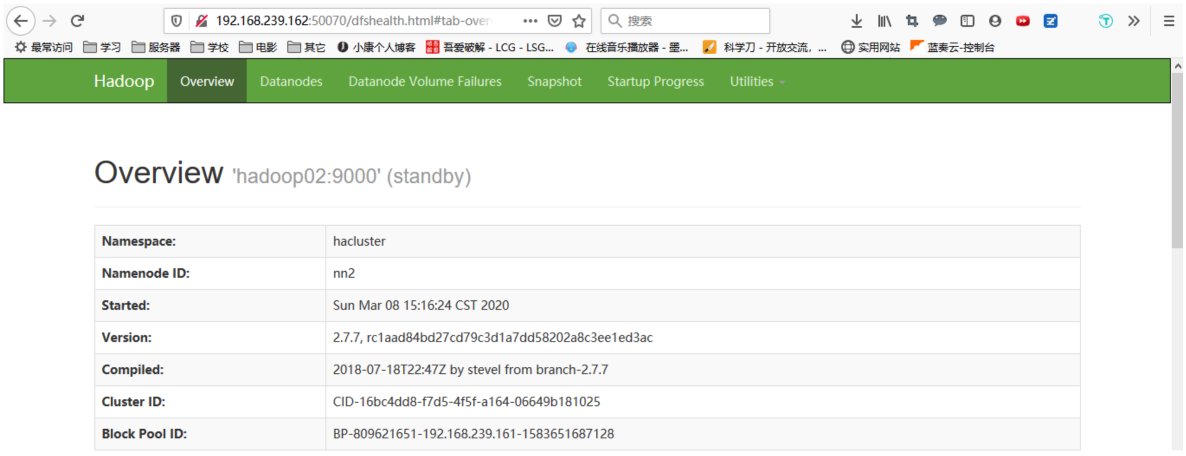
此时 hadoop01 上的 `NameNode` 处于可用状态：


192.168.239.161:50070/dfshealth.html#tab-over
搜索
最常访问 学习 服务器 学校 电影 其它 小康个人博客 吾爱破解 - LCG - LSG... 在线音乐播放器 - 墨... 科学刀 - 开放交流，... 实用网站 蓝奏云-控制台
Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

Overview 'hadoop01:9000' (active)

Namespace:	hacluster
Namenode ID:	nn1
Started:	Sun Mar 08 15:15:07 CST 2020
Version:	2.7.7, rc1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac
Compiled:	2018-07-18T22:47Z by stevel from branch-2.7.7
Cluster ID:	CID-16bc4dd8-f7d5-4f5f-a164-06649b181025
Block Pool ID:	BP-809621651-192.168.239.161-1583651687128

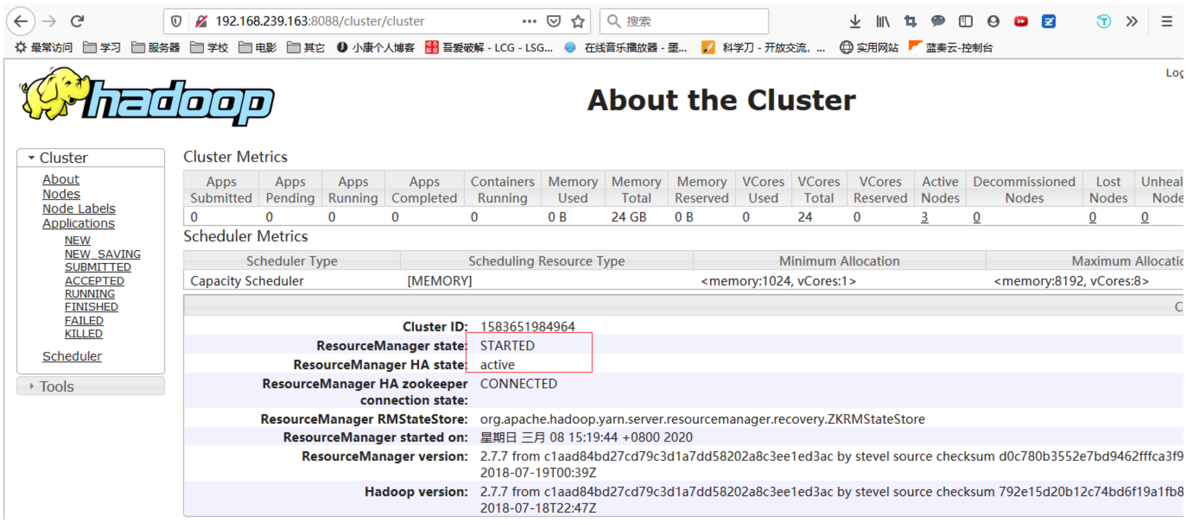
而 hadoop02 上的 `NameNode` 则处于备用状态：



Overview 'hadoop02:9000' (standby)

Namespace:	hacluster
Namenode ID:	nn2
Started:	Sun Mar 08 15:16:24 CST 2020
Version:	2.7.7, rc1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac
Compiled:	2018-07-18T22:47Z by stevel from branch-2.7.7
Cluster ID:	CID-16bc4dd8-f7d5-4f5f-a164-06649b181025
Block Pool ID:	BP-809621651-192.168.239.161-1583651687128

hadoop03 上的 `ResourceManager` 处于可用状态：



Cluster Metrics

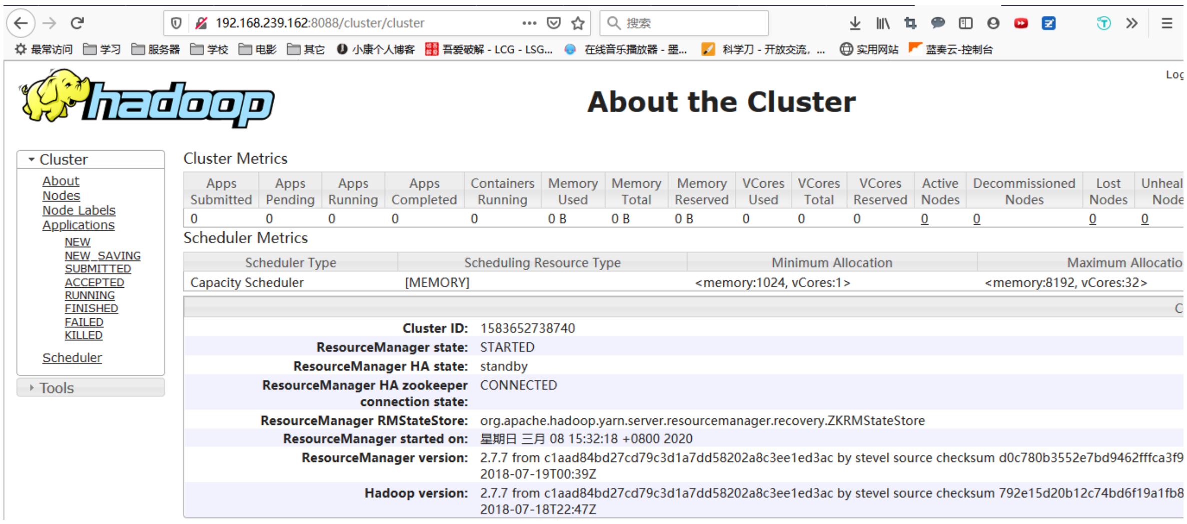
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unheal Node
0	0	0	0	0	0 B	24 GB	0 B	0	24	0	3	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocatic
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Cluster ID:	1583651984964
ResourceManager state:	STARTED
ResourceManager HA state:	active
ResourceManager HA zookeeper connection state:	CONNECTED
ResourceManager RMStateStore:	org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore
ResourceManager started on:	星期日 三月 08 15:19:44 +0800 2020
ResourceManager version:	2.7.7 from c1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac by stevel source checksum d0c780b3552e7bd9462fffca3f9 2018-07-19T00:39Z
Hadoop version:	2.7.7 from c1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac by stevel source checksum 792e15d20b12c74bd6f19a1fb8 2018-07-18T22:47Z

hadoop02 上的 `ResourceManager` 则处于备用状态：



Cluster Metrics

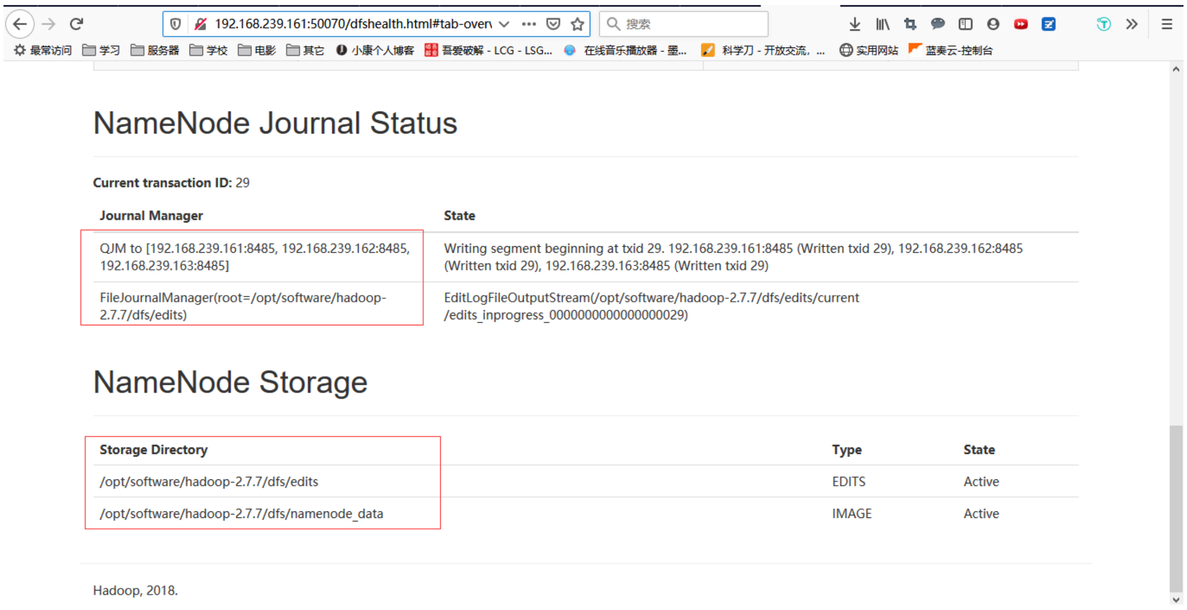
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unheal Node
0	0	0	0	0	0 B	0 B	0 B	0	0	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocatio
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:32>

Cluster ID:	1583652738740
ResourceManager state:	STARTED
ResourceManager HA state:	standby
ResourceManager HA zookeeper connection state:	CONNECTED
ResourceManager RMStateStore:	org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore
ResourceManager started on:	星期日 三月 08 15:32:18 +0800 2020
ResourceManager version:	2.7.7 from c1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac by stevel source checksum d0c780b3552e7bd9462fffca3f9 2018-07-19T00:39Z
Hadoop version:	2.7.7 from c1aad84bd27cd79c3d1a7dd58202a8c3ee1ed3ac by stevel source checksum 792e15d20b12c74bd6f19a1fb8 2018-07-18T22:47Z

同时界面上也有 `Journal Manager` 的相关信息：



NameNode Journal Status

Current transaction ID: 29

Journal Manager	State
QJM to [192.168.239.161:8485, 192.168.239.162:8485, 192.168.239.163:8485]	Writing segment beginning at txid 29. 192.168.239.161:8485 (Written txid 29), 192.168.239.162:8485 (Written txid 29), 192.168.239.163:8485 (Written txid 29)
FileJournalManager(root=/opt/software/hadoop-2.7.7/dfs/edits)	EditLogFileOutputStream(/opt/software/hadoop-2.7.7/dfs/edits/current /edits_inprogress_0000000000000000029)

NameNode Storage

Storage Directory	Type	State
/opt/software/hadoop-2.7.7/dfs/edits	EDITS	Active
/opt/software/hadoop-2.7.7/dfs/namenode_data	IMAGE	Active

Hadoop, 2018.

六、代码测试HA

```
/**
 * 测试HA集群
 *
 * @author xiaokang
 */
public class TestHDFS {
    public static void main(String[] args) throws IOException,
InterruptedException {
        Configuration conf = new Configuration();
        conf.set("fs.defaultFS", "hdfs://hacluster");
        conf.set("dfs.nameservices", "hacluster");
        conf.set("dfs.ha.namenodes.hacluster", "nn1,nn2");
        conf.set("dfs.namenode.rpc-address.hacluster.nn1",
"192.168.239.161:9000");
        conf.set("dfs.namenode.rpc-address.hacluster.nn2",
"192.168.239.162:9000");
        conf.set("dfs.client.failover.proxy.provider.hacluster",
"org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider");

        FileSystem fs = FileSystem.get(URI.create("hdfs://hacluster"), conf,
"xiaokang");
        fs.mkdirs(new Path(args[0]));
        System.out.println("ok-微信公众号：小康新鲜事儿");
    }
}
```

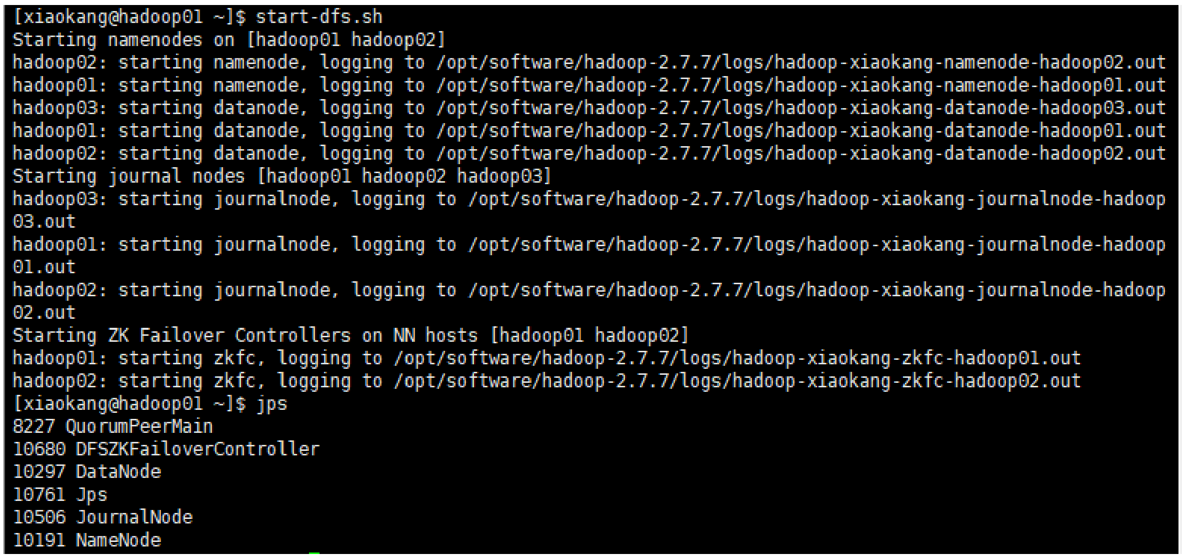
```
[xiaokang@hadoop01 ~]$ hadoop jar Zookeeper-API-1.0.jar TestHDFS /xiaokang

#杀掉active的NameNode之后，再次创建一个文件夹
[xiaokang@hadoop02 logs]$ kill -9 8277
[xiaokang@hadoop01 ~]$ hadoop jar Zookeeper-API-1.0.jar TestHDFS /xiaokang1
```

七、集群二次启动

上面的集群初次启动涉及到一些必要初始化操作，所以过程略显繁琐。但是集群一旦搭建好后，想要再次启用它是比较方便的，步骤如下（首选需要**确保 ZooKeeper 集群已经启动**）：

在 `hadoop01` 启动 HDFS，此时会启动所有与 HDFS 高可用相关的服务，包括 NameNode、DataNode 、 JournalNode和DFSZKFailoverController：

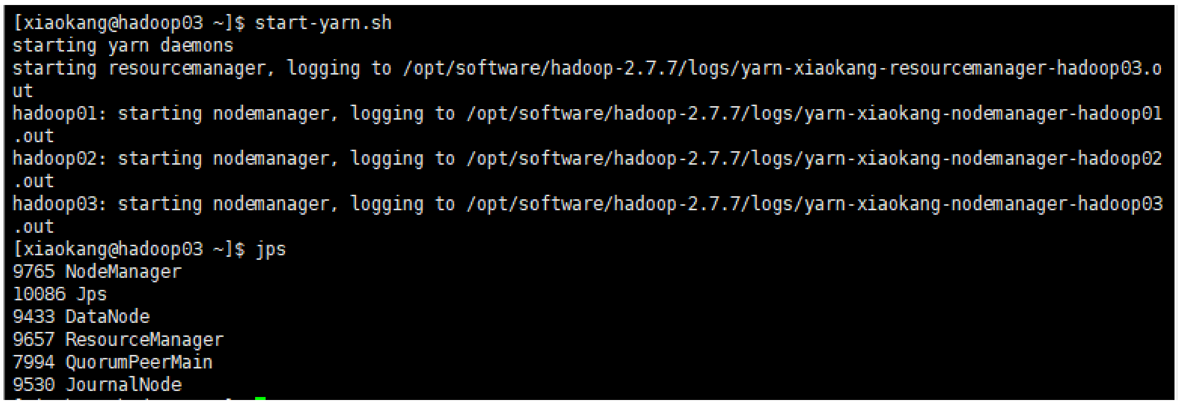
```
[xiaokang@hadoop01 ~]$ start-dfs.sh
```



```
[xiaokang@hadoop01 ~]$ start-dfs.sh
Starting namenodes on [hadoop01 hadoop02]
hadoop02: starting namenode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-namenode-hadoop02.out
hadoop01: starting namenode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-namenode-hadoop01.out
hadoop03: starting datanode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-datanode-hadoop03.out
hadoop01: starting datanode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-datanode-hadoop01.out
hadoop02: starting datanode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-datanode-hadoop02.out
Starting journal nodes [hadoop01 hadoop02 hadoop03]
hadoop03: starting journalnode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-journalnode-hadoop
03.out
hadoop01: starting journalnode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-journalnode-hadoop
01.out
hadoop02: starting journalnode, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-journalnode-hadoop
02.out
Starting ZK Failover Controllers on NN hosts [hadoop01 hadoop02]
hadoop01: starting zkfc, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-zkfc-hadoop01.out
hadoop02: starting zkfc, logging to /opt/software/hadoop-2.7.7/logs/hadoop-xiaokang-zkfc-hadoop02.out
[xiaokang@hadoop01 ~]$ jps
8227 QuorumPeerMain
10680 DFSZKFailoverController
10297 DataNode
10761 Jps
10506 JournalNode
10191 NameNode
```

在 `hadoop03` 启动 YARN：

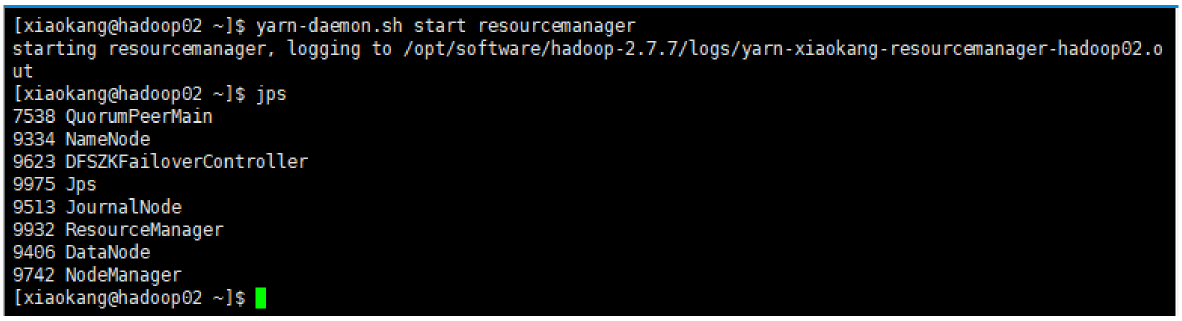
```
[xiaokang@hadoop03 ~]$ start-yarn.sh
```



```
[xiaokang@hadoop03 ~]$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /opt/software/hadoop-2.7.7/logs/yarn-xiaokang-resourcemanager-hadoop03.o
ut
hadoop01: starting nodemanager, logging to /opt/software/hadoop-2.7.7/logs/yarn-xiaokang-nodemanager-hadoop01
.out
hadoop02: starting nodemanager, logging to /opt/software/hadoop-2.7.7/logs/yarn-xiaokang-nodemanager-hadoop02
.out
hadoop03: starting nodemanager, logging to /opt/software/hadoop-2.7.7/logs/yarn-xiaokang-nodemanager-hadoop03
.out
[xiaokang@hadoop03 ~]$ jps
9765 NodeManager
10086 Jps
9433 DataNode
9657 ResourceManager
7994 QuorumPeerMain
9530 JournalNode
```

这个时候 `hadoop02` 上的 `ResourceManager` 服务通常还是没有启动的，需要手动启动：

```
[xiaokang@hadoop02 ~]$ yarn-daemon.sh start resourcemanager
```

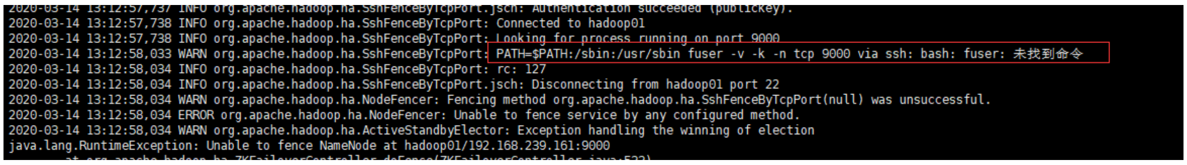


```
[xiaokang@hadoop02 ~]$ yarn-daemon.sh start resourcemanager
starting resourcemanager, logging to /opt/software/hadoop-2.7.7/logs/yarn-xiaokang-resourcemanager-hadoop02.o
ut
[xiaokang@hadoop02 ~]$ jps
7538 QuorumPeerMain
9334 NameNode
9623 DFSZKFailoverController
9975 Jps
9513 JournalNode
9932 ResourceManager
9406 DataNode
9742 NodeManager
[xiaokang@hadoop02 ~]$
```

八、踩坑分享

HA集群都启动好之后，杀掉一个active的NameNode之后，发现另一个NameNode并不能自动切换成active，而还是standby，经过查看日志发现如下错误：

```
[xiaokang@hadoop02 logs]$ tail -100 hadoop-xiaokang-zkfc-hadoop02.log
```



```
2020-03-14 13:12:57,737 INFO org.apache.hadoop.ha.SshFenceByTcpPort.jsch: Authentication succeeded (publickey).
2020-03-14 13:12:57,738 INFO org.apache.hadoop.ha.SshFenceByTcpPort: Connected to hadoop01
2020-03-14 13:12:57,738 INFO org.apache.hadoop.ha.SshFenceByTcpPort: Looking for process running on port 9000
2020-03-14 13:12:58,033 WARN org.apache.hadoop.ha.SshFenceByTcpPort: PATH=$PATH:/sbin:/usr/sbin fuser -v -k -n tcp 9000 via ssh: bash: fuser: 未找到命令
2020-03-14 13:12:58,034 INFO org.apache.hadoop.ha.SshFenceByTcpPort: rc: 127
2020-03-14 13:12:58,034 INFO org.apache.hadoop.ha.SshFenceByTcpPort.jsch: Disconnecting from hadoop01 port 22
2020-03-14 13:12:58,034 WARN org.apache.hadoop.ha.NodeFencer: Fencing method org.apache.hadoop.ha.SshFenceByTcpPort(null) was unsuccessful.
2020-03-14 13:12:58,034 ERROR org.apache.hadoop.ha.NodeFencer: Unable to fence service by any configured method.
2020-03-14 13:12:58,034 WARN org.apache.hadoop.ha.ActiveStandbyElector: Exception handling the winning of election
java.lang.RuntimeException: Unable to fence NameNode at hadoop01/192.168.239.161:9000
        at org.apache.hadoop.ha.ZKFailoverController.doFence(ZKFailoverController.java:533)
```

这个错误就是找不到fuser命令，原因就是我机器没有安装`psmisc`

解决方法：（NameNode节点上安装上psmisc即可）

```
[xiaokang@hadoop01 ~]$ sudo yum -y install psmisc
[xiaokang@hadoop02 ~]$ sudo yum -y install psmisc
```