

Hadoop 集群搭建

目录

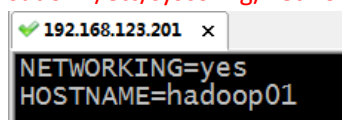
1、基础集群环境准备.....	1
1.1、修改主机名.....	1
1.2、设置系统默认启动级别.....	2
1.3、配置 hadoop 用户 sudoer 权限.....	2
1.4、配置 IP	2
1.5、关闭防火墙/关闭 Selinux.....	3
1.6、添加内网域名映射.....	3
1.7、安装 JDK.....	3
1.8、同步服务器时间.....	4
1.9、配置免密登录.....	4
2、Hadoop 集群环境安装.....	6
2.1、Hadoop 版本选择.....	6
2.2、安装 hadoop	6
2.2.1 hadoop 伪分布式模式安装	6
2.3.2 hadoop 分布式集群安装	8
3、集群初步使用.....	11
3.1、Hadoop 集群启动.....	11
3.2、HDFS 初步使用	11
3.3、mapreduce 初步使用	11
4、hadoop 集群安装高级知识	11
4.1、Hadoop HA 安装	11
4.2、Hadoop 配置机架感知.....	12
4.3、Hadoop Fedaration	13

1、基础集群环境准备

1.1、修改主机名

- 在 root 账号下用命令：**vi /etc/sysconfig/network**
或者如果配置了 hadoop sudo 权限，则在 hadoop 登录情况下使用命令：

sudo vi /etc/sysconfig/network



- 修改好后，保存退出即可

1.2、设置系统默认启动级别

1、在 root 账号下输入 **vi /etc/inittab**

```
hadoop@localhost:/home/had
File Edit View Search Terminal Help
[root@localhost Desktop]# vi /etc/inittab
```

2、改默认启动级别，3 是多用户模式，并且不启动图形界面

```
# Default runlevel. The runlevels used are:
# 0 - halt (Do NOT set initdefault to this)
# 1 - Single user mode
# 2 - Multiuser, without NFS (The same as 3, if you do not have networking)
# 3 - Full multiuser mode
# 4 - unused
# 5 - X11
# 6 - reboot (Do NOT set initdefault to this)
# 这里默认是5，改成3，保存
id:3:initdefault:
```

1.3、配置 hadoop 用户 sudoer 权限

1、在 root 账号下，命令终端输入：**vi /etc/sudoers**

```
File Edit View Search Terminal Help
[root@localhost Desktop]# vi /etc/sudoers
```

2、找到

```
root    ALL=(ALL)    ALL
```

这一行，然后在他下面添加一行：

```
hadoop  ALL=(ALL)    ALL
```

保存，退出

```
hadoop@localhost:/home/hadoop/Desktop
File Edit View Search Terminal Help
##
## The COMMANDS section may have other options added to it.
##
## Allow root to run any commands anywhere
root    ALL=(ALL)    ALL
hadoop  ALL=(ALL)    ALL
```

1.4、配置 IP

Linux 服务器的 IP 修改方式有三种，在此不细讲，请参考文档：[资料-linux 修改 IP 三种方式.pdf](#)

1.5、关闭防火墙/关闭 Selinux

防火墙操作相关:

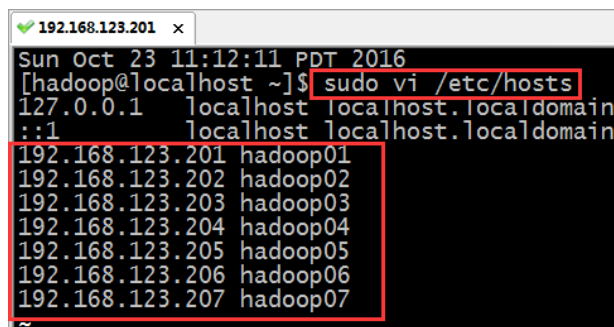
查看防火墙状态: service iptables status
关闭防火墙: service iptables stop
开启防火墙: service iptables start
重启防火墙: service iptables restart
关闭防火墙开机启动: chkconfig iptables off
开启防火墙开机启动: chkconfig iptables on

关闭 Selinux: 具体做法是修改/etc/selinux/config 配置文件中的 **SELINUX=disabled**

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#   targeted - Targeted processes are protected,
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

1.6、添加内网域名映射

修改配置文件: vi /etc/hosts



```
192.168.123.201 x
Sun Oct 23 11:12:11 PDT 2016
[hadoop@localhost ~]$ sudo vi /etc/hosts
127.0.0.1    localhost localhost.localdomain
::1         localhost localhost.localdomain
192.168.123.201 hadoop01
192.168.123.202 hadoop02
192.168.123.203 hadoop03
192.168.123.204 hadoop04
192.168.123.205 hadoop05
192.168.123.206 hadoop06
192.168.123.207 hadoop07
```

1.7、安装 JDK

- 1、上传 jdk-8u73-linux-x64.tar.gz
- 2、解压到/usr/local 目录下
tar -zxvf jdk-8u73-linux-x64.tar.gz -C /usr/local
- 3、配置环境变量
 - a) vi /etc/profile
 - b) 在最后加入两行:
export JAVA_HOME=/usr/local/jdk1.8.0_73

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

c) 保存退出

4、source /etc/profile

5、检测是否安装成功，输入命令：java -version

做完以上步骤后，可以开始克隆虚拟机。因为以上系统的配置，都是一些基础性的操作。都是必须的。

1.8、同步服务器时间

1、使用 date 命令手动简单同步一下

命令：**date -s "2016-10-23 11:11:11"**

修改时间后，需要写入硬件 bios 才能在重启之后依然生效

命令：hwclock -w

2、配置 crontab 服务，用 ntpdate 定时同步时间（推荐方式）

ntpdate 202.120.2.101

3、如果类似 202.120.2.101 这种网络时间服务器无法访问，那么请自行搭建时间服务器

以上两种方式不管怎么做，都**不要忘记更改时区**

1.9、配置免密登录

Linux 配置免密登录有两种方式：

3、第一种

a) 在 hadoop 登录状态下，输入命令 ssh-keygen 或者 ssh-keygen -t rsa

b) 之后你会发现，在/home/hadoop/.ssh 目录下生成了公钥文件

```
[hadoop@hadoop04 .ssh]$ pwd
/home/hadoop/.ssh
[hadoop@hadoop04 .ssh]$ ll
total 8
-rw-----. 1 hadoop hadoop 1671 Nov  1 14:16 id_rsa
-rw-r--r--. 1 hadoop hadoop  397 Nov  1 14:16 id_rsa.pub
[hadoop@hadoop04 .ssh]$
```

c) 复制公钥文件到授权列表

```
cat ./id_rsa.pub >> authorized_keys
```

d) 修改文件权限

```
chmod 600 ./authorized_keys
```

```
[hadoop@hadoop04 .ssh]$ pwd
/home/hadoop/.ssh
[hadoop@hadoop04 .ssh]$ ll
total 8
-rw----- 1 hadoop hadoop 1671 Nov 1 14:16 id_rsa
-rw-r--r-- 1 hadoop hadoop 397 Nov 1 14:16 id_rsa.pub
[hadoop@hadoop04 .ssh]$ cat ./id_rsa.pub >> authorized_keys
[hadoop@hadoop04 .ssh]$ ll
total 12
-rw-rw-r-- 1 hadoop hadoop 397 Nov 1 14:23 authorized_keys
-rw----- 1 hadoop hadoop 1671 Nov 1 14:16 id_rsa
-rw-r--r-- 1 hadoop hadoop 397 Nov 1 14:16 id_rsa.pub
[hadoop@hadoop04 .ssh]$ chmod 600 ./authorized_keys
[hadoop@hadoop04 .ssh]$ ll
total 12
-rw----- 1 hadoop hadoop 397 Nov 1 14:23 authorized_keys
-rw----- 1 hadoop hadoop 1671 Nov 1 14:16 id_rsa
-rw-r--r-- 1 hadoop hadoop 397 Nov 1 14:16 id_rsa.pub
[hadoop@hadoop04 .ssh]$
```

- e) 将该授权文件 authorized_keys 文件复制到 slave 节点

scp ./authorized_keys [hadoop@hadoop02:~/.ssh/](#)

```
[hadoop@hadoop04 .ssh]$ scp ./authorized_keys hadoop@hadoop02:~/.ssh/
The authenticity of host 'hadoop02 (192.168.123.202)' can't be established.
RSA key fingerprint is 73:cb:0c:7b:94:63:36:0e:ba:f3:84:a7:4e:41:39:9d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'hadoop02,192.168.123.202' (RSA) to the list of known hosts
hadoop@hadoop02's password:
authorized_keys
[hadoop@hadoop04 .ssh]$ ssh hadoop02
Last login: Tue Nov 1 13:52:59 2016 from 192.168.123.11
[hadoop@hadoop02 ~]$ exit
logout
Connection to hadoop02 closed.
[hadoop@hadoop04 .ssh]$
```

- f) 检查免密登录是否设置成功

ssh hadoop02

看看是不是能登录进入 hadoop02 的服务器

4、第二种

- a) 在 hadoop 登录状态下, 输入命令 ssh-keygen 或者 ssh-keygen -t rsa
b) 之后你会发现, 在 /home/hadoop/.ssh 目录下生成了公钥文件

```
[hadoop@hadoop04 .ssh]$ pwd
/home/hadoop/.ssh
[hadoop@hadoop04 .ssh]$ ll
total 8
-rw----- 1 hadoop hadoop 1671 Nov 1 14:16 id_rsa
-rw-r--r-- 1 hadoop hadoop 397 Nov 1 14:16 id_rsa.pub
[hadoop@hadoop04 .ssh]$
```

- c) 使用一个更简单的方式, 使用命令:

ssh-copy-id hadoop02

建立 hadoop01 到 hadoop02 的免密登录

```
[hadoop@hadoop01 ~]$ ssh-copy-id hadoop02
The authenticity of host 'hadoop02 (192.168.123.202)' can't be established.
RSA key fingerprint is 73:cb:0c:7b:94:63:36:0e:ba:f3:84:a7:4e:41:39:9d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'hadoop02,192.168.123.202' (RSA) to the list of known hosts.
hadoop@hadoop02's password:
Now try logging into the machine, with "ssh 'hadoop02'", and check in:

  .ssh/authorized_keys
to make sure we haven't added extra keys that you weren't expecting.
[hadoop@hadoop01 ~]$ ssh hadoop02
Last login: Tue Nov 1 14:30:26 2016 from hadoop04
[hadoop@hadoop02 ~]$ exit
logout
Connection to hadoop02 closed.
[hadoop@hadoop01 ~]$
```

2、Hadoop 集群环境安装

2.1、Hadoop 版本选择

1、apache 官方版本：

大版本：0.20.2

1.X

2.X

2.5.2

2.6.5

2.7.3

2、商业发型版本：

提供完善的管理系统，修复 bug 可能会领先于官方版本

Cloudera 公司的 CDH：5.7.X

2.2、安装 hadoop

2.2.1 hadoop 伪分布式模式安装

Hadoop 可以在单节点上以伪分布式的方式运行，Hadoop 进程以分离的 Java 进程来运行，节点既作为 NameNode 也作为 DataNode，同时，读取的是 HDFS 中的文件。

Hadoop 的配置文件位于 hadoop-2.6.5/etc/hadoop/ 文件夹中，伪分布式需要修改 2 个配置文件 core-site.xml 和 hdfs-site.xml（其中 hdfs-site.xml 主要用来配置数据块的副本数的，对于伪分布式来说，不管你配置几个副本，它始终都只有一个副本，所以就不用管了）。Hadoop 的配置文件是 xml 格式，每个配置以声明 property 的 name 和 value 的方式来实现。

1、修改 hadoop-env.sh 配置文件，添加 jdk 安装目录配置

```
[hadoop@hadoop01 hadoop]$ vi hadoop-env.sh
```

```
# The java implementation to use.  
export JAVA_HOME=/usr/local/jdk1.8.0_73
```

2、修改 core-site.xml

```
<configuration>  
  <property>  
    <name>hadoop.tmp.dir</name>  
    <value>/home/hadoop/hadoopdata</value>  
    <description>Abase for other temporary directories.</description>  
  </property>  
  <property>  
    <name>fs.defaultFS</name>  
    <value>hdfs://hadoop01:9000</value>  
  </property>  
</configuration>
```

5、修改 slaves

```
hadoop01
```

6、添加 hadoop 环境变量

```
export HADOOP_HOME=/home/hadoop/apps/hadoop-2.6.5
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

7、格式化 namenode

```
[hadoop@hadoop01 ~]$ hadoop namenode -format
```

```
16/11/01 15:14:09 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
16/11/01 15:14:09 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry
s 600000 millis
16/11/01 15:14:09 INFO util.GSet: Computing capacity for map NameNodeRetryCache
16/11/01 15:14:09 INFO util.GSet: VM type = 64-bit
16/11/01 15:14:09 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
16/11/01 15:14:09 INFO util.GSet: capacity = 2^15 = 32768 entries
16/11/01 15:14:09 INFO namenode.NNConf: ACLs enabled? false
16/11/01 15:14:09 INFO namenode.NNConf: XAttrs enabled? true
16/11/01 15:14:09 INFO namenode.NNConf: Maximum size of an xattr: 16384
16/11/01 15:14:10 INFO namenode.ESImage: Allocated new BlockPoolId: BP-1008002739-192.168.123.201-1478038449858
16/11/01 15:14:10 INFO common.Storage: Storage directory /home/hadoop/temp/dfs/name has been successfully formatted
16/11/01 15:14:10 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with Exid >= 0
16/11/01 15:14:10 INFO util.ExitUtil: Exiting with status 0
16/11/01 15:14:10 INFO namenode.NameNode: SHUTDOWN_MSG:
*****
SHUTDOWN_MSG: Shutting down NameNode at hadoop01/192.168.123.201
*****
[hadoop@hadoop01 ~]$
```

最后出现如红线所示提示，则表示格式化成功

8、启动 hdfs

```
3687 Jps
[hadoop@hadoop01 .ssh]$ start-dfs.sh
16/11/01 15:27:44 WARN util.NativeCodeLoader: Unable to load
a classes where applicable
Starting namenodes on [hadoop01]
hadoop01: starting namenode, logging to /home/hadoop/hadoop-2
localhost: starting datanode, logging to /home/hadoop/hadoop-
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/hadoop/
ut
16/11/01 15:28:02 WARN util.NativeCodeLoader: Unable to load
a classes where applicable
[hadoop@hadoop01 .ssh]$ jps
4165 Jps
3794 NameNode
4062 SecondaryNameNode
3911 DataNode
[hadoop@hadoop01 .ssh]$
```

9、启动 yarn

```
3911 DataNode
[hadoop@hadoop01 .ssh]$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/hadoop/hadoop-2
localhost: starting nodemanager, logging to /home/hadoop/hadoop-2
[hadoop@hadoop01 .ssh]$ jps
4302 NodeManager
3794 NameNode
4211 ResourceManager
4062 SecondaryNameNode
4516 Jps
3911 DataNode
[hadoop@hadoop01 .ssh]$
```

10、检查集群启动是否成功

- 1、利用 jps 工具检测各进程是否启动成功
- 2、Web UI 查看

HDFS: <http://hadoop01:50070/>

MapReduce: <http://hadoop01:8088/cluster/cluster>

2.3.2 hadoop 分布式集群安装

集群规划:

	HDFS	YARN
Hadoop02	NameNode + DataNode 主节点	NodeManager
Hadoop03	DataNode + SecondaryNamenode	NodeManager
Hadoop04	DataNode	NodeManager + ResourceManager 主节点

总共三个 datanode 节点, 设置副本数为 2, 是为了观察数据块分布方便, 大家可以根据自己的机器性能酌情决定

具体步骤:

一、上传安装包, 并解压到相关目录

```
put c:/hadoop-2.6.5-centos-6.7.tar.gz
tar -zxvf hadoop-2.6.5-centos-6.7.tar.gz -C /home/hadoop/apps
```

二、主要讲配置文件的配置

1、修改 hadoop-env.sh

```
export JAVA_HOME=/usr/java/jdk1.8.0_73
```

2、修改 core-site.xml

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://hadoop02:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/hadoop/hadoopdata</value>
</property>
```

3、修改 hdfs-site.xml

```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/home/hadoop/hadoopdata/name</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>/home/hadoop/hadoopdata/data</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>2</value>
```



```
</property>
<property>
  <name>dfs.secondary.http.address</name>
  <value>hadoop03:50090</value>
</property>
```

4、修改 mapred-site.xml（集群只有 mapred-site.xml.template，可以从这个文件进行复制，或者直接改名也可）

```
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
```

5、修改 yarn-site.xml

```
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>hadoop04</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
```

6、修改 slaves 文件，添加以下内容，该内容是从节点列表

```
hadoop02
hadoop03
hadoop04
```

三、分发安装包到各个节点，Hadoop 集群的每个节点都需要安装 Hadoop 安装包

```
scp -r hadoop-2.6.4 hadoop02:$PWD
scp -r hadoop-2.6.4 hadoop03:$PWD
```

四、在 HDFS 主节点上执行命令进行初始化 namenode

```
bin/hadoop namenode -format
```

查看最后是否初始化成功

```
17/04/25 04:04:59 INFO util.GSet: Computing capacity for map NameNodeRetryCache
17/04/25 04:04:59 INFO util.GSet: VM type = 64-bit
17/04/25 04:04:59 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
17/04/25 04:04:59 INFO util.GSet: capacity = 2^15 = 32768 entries
17/04/25 04:04:59 INFO namenode.NNConf: ACLs enabled? false
17/04/25 04:04:59 INFO namenode.NNConf: XAttrs enabled? true
17/04/25 04:04:59 INFO namenode.NNConf: Maximum size of an xattr: 16384
17/04/25 04:04:59 INFO namenode.FSImage: Alllocated new blockpoolid: BP-1240124751-192.168.123.102-1493064290810
17/04/25 04:04:59 INFO namenode.Storage: Storage directory /home/hadoop/hadoopdata/name has been successfully formatted.
17/04/25 04:04:59 INFO namenode.FSImageFormatProtobuf: Saving image file /home/hadoop/hadoopdata/name/current/fsimage.ckpt
t.000000000000000000 using no compression
17/04/25 04:05:00 INFO namenode.FSImageFormatProtobuf: Image file /home/hadoop/hadoopdata/name/current/fsimage.ckpt_00000
000000000000000 of size 322 bytes saved in 0 seconds.
17/04/25 04:05:00 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
17/04/25 04:05:00 INFO util.ExitUtil: Exiting with status 0
17/04/25 04:05:00 INFO namenode.NameNode: SHUTDOWN_MSG:
/*
*****
SHUTDOWN_MSG: Shutting down NameNode at hadoop02/192.168.123.102
*****
*/
```

五、在 HDFS 上的主节点启动 HDFS，其实在哪里启动并无关系

sbin/start-dfs.sh

结果：在主节点启动了 NameNode 守护进程
在从节点启动了 DataNode 守护进程
在配置的一个特定节点上 hadoop03 会启动 SecondaryNameNode 进程

```

hadoop02 [hadoop@hadoop02 hadoopdata]$ jps
6338 Jps
5890 NameNode
6229 NodeManager
6007 DataNode
[hadoop@hadoop02 hadoopdata]$

hadoop03 [hadoop@hadoop03 ~]$ jps
3648 NodeManager
3761 Jps
3572 SecondaryNameNode
3484 DataNode
[hadoop@hadoop03 ~]$

hadoop04 [hadoop@hadoop04 ~]$ jps
18336 ResourceManager
18738 Jps
18227 DataNode
18428 NodeManager
[hadoop@hadoop04 ~]$
  
```

六、在 YARN 主节点启动 YARN，要求在 YARN 主节点进行启动，否则 ResourceManager 主进程会启动不成功，需要额外手动启动

sbin/start-yarn.sh

结果：在主节点启动了 resourcemanager 守护进程
在从节点启动了 nodemanager 守护进程

七、测试集群是否安装成功

1、检测 hdfs 是否启动成功

hadoop fs -ls /

2、检测 yarn 集群是否启动成功（提交 mapreduce 例子程序试跑）

bin/hadoop jar hadoop-mapreduce-examples-2.6.4.jar pi 5 5

八、补充说明

1、启动 namenode 或者 datanode

sbin/hadoop-daemon.sh start datanode

sbin/hadoop-daemon.sh start namenode

2、启动 yarn nodemanager

sbin/yarn-daemon.sh start nodemanager

sbin/yarn-daemon.sh start resourcemanager

3、hdfs 集群信息 web 管理界面地址

http://hadoop02:50070

4、mapreduce 运行状态信息 web 管理界面

http://hadoop04:8088

5、查看集群状态

hadoop dfsadmin -report

hdfs dfsadmin -report（推荐用这个）

3、集群初步使用

3.1、Hadoop 集群启动

DFS 集群启动: `sbin/start-dfs.sh`

DFS 集群关闭: `sbin/stop-dfs.sh`

YARN 集群启动: `sbin/start-yarn.sh`

YARN 集群关闭: `sbin/stop-yarn.sh`

3.2、HDFS 初步使用

查看集群文件: `hadoop fs -ls /`

上传文件: `hadoop fs -put filepath destpath`

下载文件: `hadoop fs -get destpath`

创建文件夹: `hadoop fs -mkdir /hadoopdata`

查看文件内容: `hadoop fs -cat /hadoopdata/mysecret.txt`

3.3、mapreduce 初步使用

演示 hadoop 自带的例子程序并观察执行过程:

1、求 PI

```
hadoop jar hadoop-mapreduce-examples-2.6.5.jar pi 5 5
```

(该 jar 位于: `/home/hadoop/apps/hadoop-2.6.5/share/hadoop/mapreduce/`)

执行结果:

Job Finished in 70.913 seconds

Estimated value of Pi is 3.20000000000000000000

2、单词计数

```
hadoop jar hadoop-mapreduce-examples-2.6.5.jar wordcount /hadoopdata/mysecret.txt
```

4、hadoop 集群安装高级知识

4.1、Hadoop HA 安装

为什么会有 hadoop HA 机制呢?

HA: High Available, 高可用

在 Hadoop 2.0 之前,在 HDFS 集群中 NameNode 存在单点故障 (SPOF: A Single Point of Failure)。

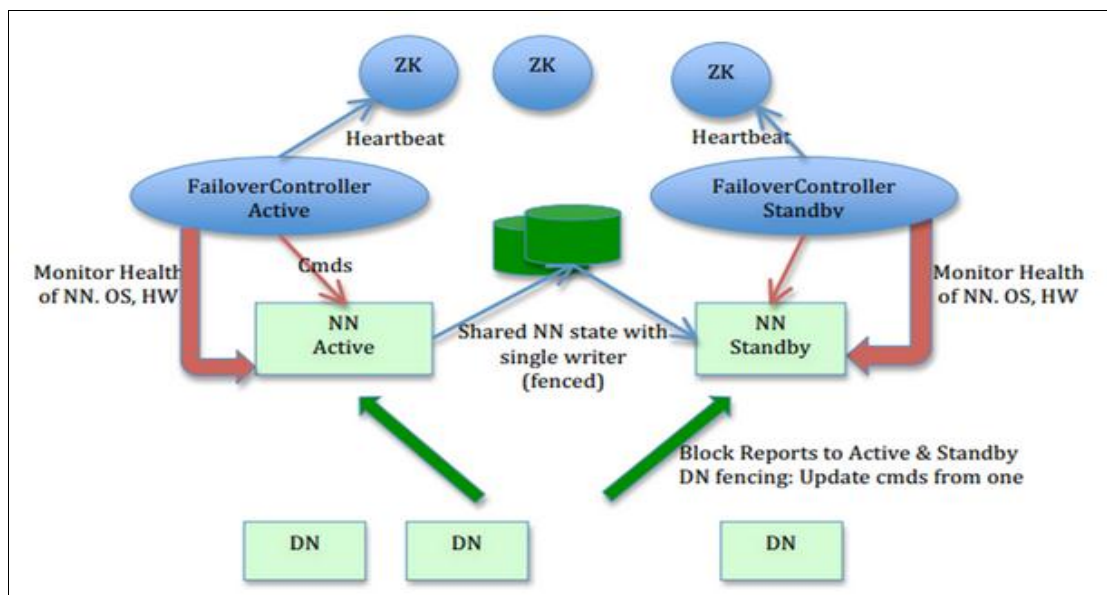
对于只有一个 NameNode 的集群，如果 NameNode 机器出现故障(比如宕机或是软件、硬件升级)，那么整个集群将无法使用，直到 NameNode 重新启动

那如何解决呢？

HDFS 的 HA 功能通过配置 Active/Standby 两个 NameNodes 实现在集群中对 NameNode 的热备来解决上述问题。如果出现故障，如机器崩溃或机器需要升级维护，这时可通过此种方式将 NameNode 很快的切换到另外一台机器。

在一个典型的 HDFS(HA) 集群中，使用两台单独的机器配置为 NameNodes。在任何时间点，确保 NameNodes 中只有一个处于 Active 状态，其他的处在 Standby 状态。其中 ActiveNameNode 负责集群中的所有客户端操作，StandbyNameNode 仅仅充当备机，保证一旦 ActiveNameNode 出现问题能够快速切换。

为了能够实时同步 Active 和 Standby 两个 NameNode 的元数据信息（实际上 editlog），需提供一个共享存储系统，可以是 NFS、QJM（Quorum Journal Manager）或者 Zookeeper，Active Namenode 将数据写入共享存储系统，而 Standby 监听该系统，一旦发现有新数据写入，则读取这些数据，并加载到自己内存中，以保证自己内存状态与 Active NameNode 保持基本一致，如此这般，在紧急情况下 standby 便可快速切为 active namenode。为了实现快速切换，Standby 节点获取集群的最新文件块信息也是很有必要的。为了实现这一目标，DataNode 需要配置 NameNodes 的位置，并同时给他们发送文件块信息以及心跳检测。



思考问题：SecondaryNameNode 和 Standby Namenode 的区别？

4.2、Hadoop 配置机架感知

集群网络拓扑图

机架感知配置演示

<http://blog.csdn.net/l1028386804/article/details/51935169>

4.3、Hadoop Federation

为什么要有 Federation 机制呢？

在 Hadoop 2.0 之前，HDFS 的单 NameNode 设计带来很多问题，包括单点故障、内存受限，制约集群扩展性和缺乏隔离机制（不同业务使用同一个 NameNode 导致业务相互影响）等。为了解决这些问题，除了用基于共享存储的 HA 解决方案，我们还可以用 HDFS 的 Federation 机制来解决这个问题。

什么是 Federation 机制？

HDFS Federation 是指 HDFS 集群可同时存在多个 NameNode。这些 NameNode 分别管理一部分数据，且共享所有 DataNode 的存储资源。

这种设计可解决单 NameNode 存在的以下几个问题：

- 1、 HDFS 集群扩展性。多个 NameNode 分管一部分目录，使得一个集群可以扩展到更多节点，不再像 1.0 中那样由于内存的限制制约文件存储数目。
- 2、 性能更高效。多个 NameNode 管理不同的数据，且同时对外提供服务，将为用户提供更高的读写吞吐率。
- 3、 良好的隔离性。用户可根据需要将不同业务数据交由不同 NameNode 管理，这样不同业务之间影响很小。

注意问题：HDFS Federation 并不能解决单点故障问题，也就是说，每个 NameNode 都存在在单点故障问题，你需要为每个 namenode 部署一个 backup namenode 以应对 NameNode 挂掉对业务产生的影响。