**hadoop学习笔记**

官网：http://hadoop.apache.org

hadoop是什么？

hadoop是分布式文件系统+计算框架，面向大数据处理(按当前单台硬件和软件无法处理的文件数据)，擅长离线数据分析，是一个快速进化、发展的生态系统，应用领域通常在产生大量数据或日志的行业，例如淘宝、京东、腾讯、百度、电信运营商等，用来分析用户行为。当然也可以用在超级市场，在购物车上安装传感器，可以记录客户行走轨迹、逗留时间、商品种类，分析数据来判断商品摆放位置合不合理。

# 1 hadoop起源

hadoop思想来源是google搜索引擎搜索系统。

google搜索引擎服务器特点：

(1) 不用超级计算机，不用硬盘存储，大量使用普通的PC服务器(去掉机箱，外设、硬盘)。

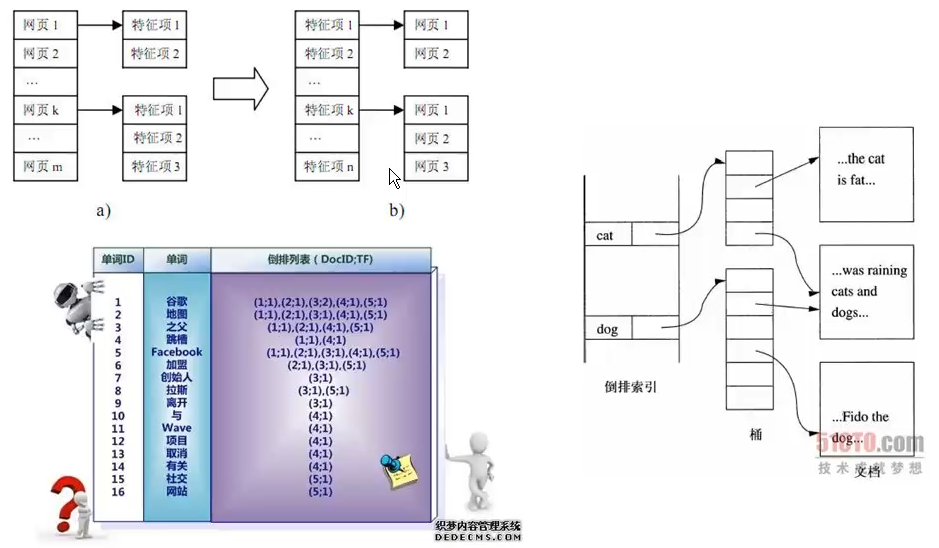
(2) 把所有爬虫收集回来的网页存储在内存，使得访问网页速度非常快，当然网页不止存储在一台服务器节点上，会存储在多台服务器节点（冗余节点），即使一台服务器坏了，也可以从其他节点自动检索出来网页。

(3) google使用倒排序索引技术，倒排序索引技术能够在极短时间内通过搜索关键词定位到包含这些关键词的网页，

常规的索引是文档到关键词的映射(文档-->关键词)，但是这样检索关键词的时候很费力，要把所有文档从头到尾去搜索，费时费劲，当文档非常多时，体验非常的差。

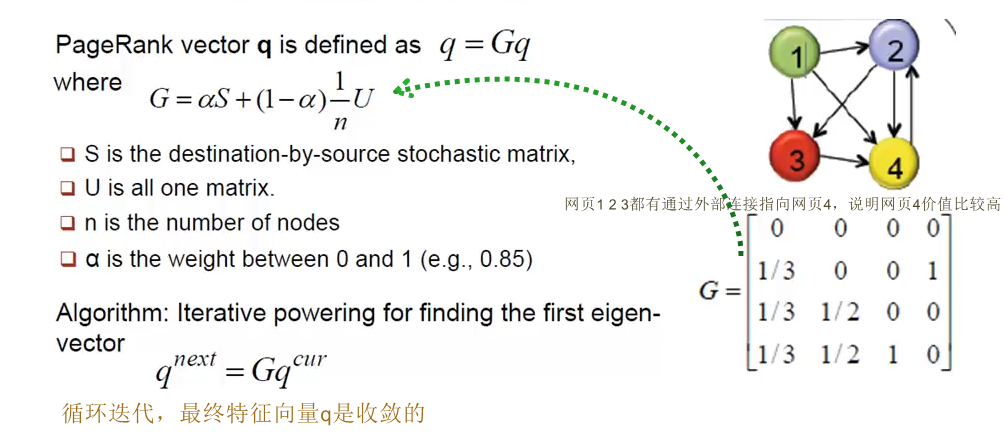
后来人们发明了倒排索引，倒排索引是关键词到文档的映射(关键词-->文档)，只要有关键词，立马就能找到她在那个文档里出现过。(过程：通过哈希索引计算关键词对应数据库的哪一行，改行已经事先存储关键词和文档，这样就通过关键词早到文档了)

原理如下：



(4) google使用核心算法Page-Rank(给每个网页进行价值评分)，通过关键词搜索回来一大堆网页，绝大多数都不是用户想要的内容，通过Page-Rank算法把用户想获取的内容优先显示给用户，使得用户体验非常好。这就是google把垃圾数据提炼成金的核心算法。

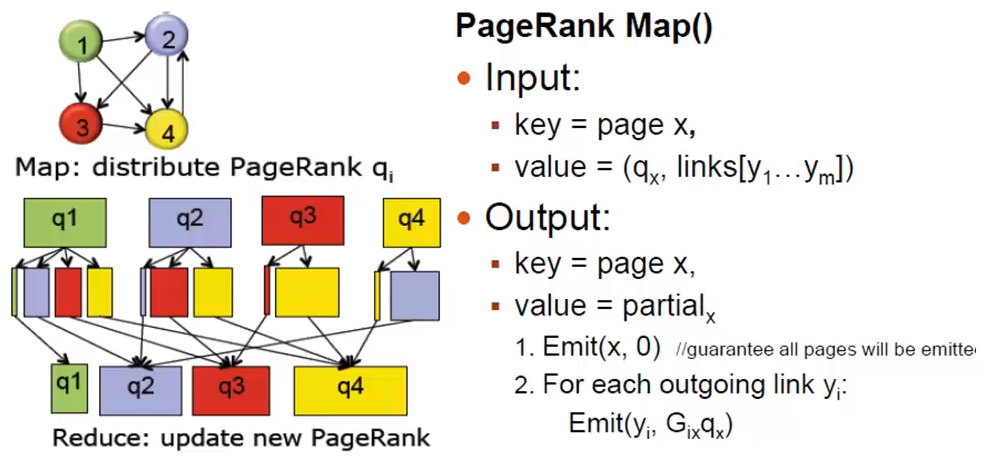
算法原理：

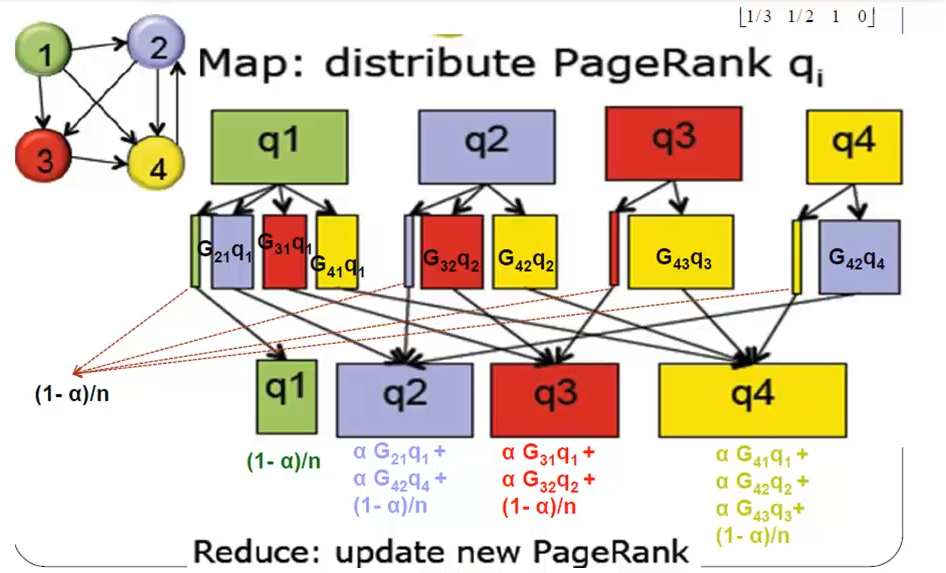


从数学上可以计算出最终特征向量q，但是当网页数庞大的时候，对于当前一台计算机来说是一个非常耗时的计算过程，内存也存放不下这么庞大数据。

解决办法：map-reduce思想

map-reduce思想是把一个庞大的计算任务，通过map映射到各个节点来进行分布式计算，每个节点只要完成一小段计算任务，最后所有节点把计算结果送到一个节点去汇总，这个汇总的过程叫reduce，原理如下：





google的关键技术和思想，也是hadoop的关键技术核心和思想。

GFS(google文件系统，通过冗余方式把网页存储到各个服务器内存中)

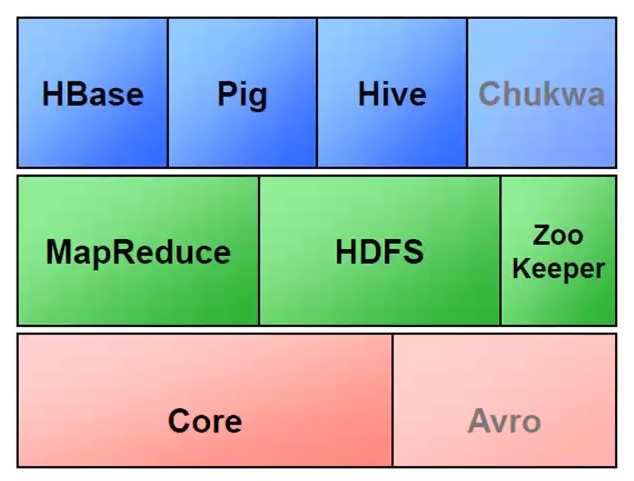
Map-Reduce(对网页进行价值评分)

Bigtable(超级大数据表，把所有数据放在一个表结构里，简化查询)

# 2 hadoop架构

## 2.1hadoop子项目家族

如下图所示：

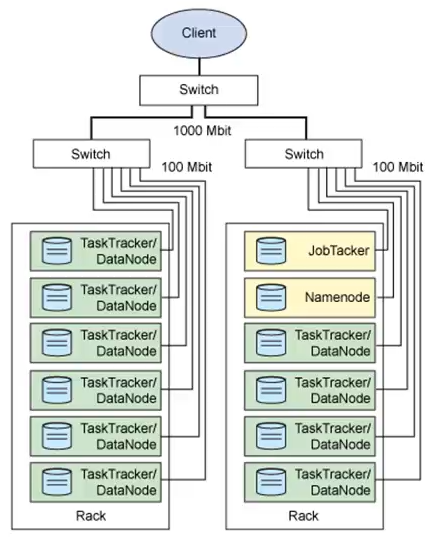
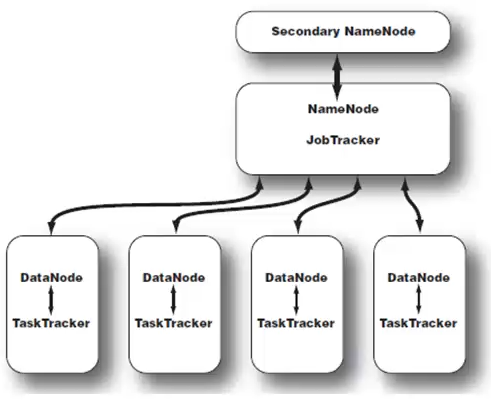


hadoop子项目命名比较有意思，大多都是用动物名称来命名的。

|  |  |
| --- | --- |
| **名称** | **说明** |
| **Core** | hadoop的核心代码 |
| **MapReduce** | hadoop核心支柱 |
| **HDFS** | hadoop文件系统 |
| **ZooKeeper** | 负责节点和进程之间的通信，是通信的协调工具 |
| **Pig** | 轻量级语言，用户到MapReduce的转换器，类似于linux系统下shell语言作用。用户输入命令，pig把命令转换为MapReduce执行程序，然后把结果返回给用户。产生pig子模块的原因是hadoop是用java写的，但不是所有用户都会java，所以通过pig子模块来给不会java人员操作hadoop |
| **Hive** | 是sql语言到hadoop的映射器。用户输入sql语言，Hive自动转为MapReduce，MapReduce把处理结果返回用户 |
| **Hbase** | 属于nosql数据库，Hbase数据库最大特点是按列来存储数据的，好处就是提高检索速度和减少IO的量 |
| **Chukwa** | 数据集成工具。例如现场生成的日志信息，Chukwa能够定时去抓取日志信息过来进行数据分析 |

## 2.2 hadoop的进程

一个集群hadoop的进程分布，如下图所示



|  |  |
| --- | --- |
| 进程名称 | 说明 |
| **Namenode** | 名称节点，它的特点：  (1) 是HDFS的守护进程，起到分布式文件存储的总控作用  (2) 记录所有的源数据，记录文件是如何分割成数据块的，并且记录这些数据块存储到哪些节点哪个数据块上。  (3) 对内存IO进行集中管理，当用户和hadoop交互时，首先要通过Namenode进程来获得文件分布存储状态的信息，然后才能从数据节点获取数据。  (4) 是个单点，发生故障时将使得整个集群崩溃 |
| **Secondary Namenode** | 辅助名称节点。把Namenode的源数据信息重新保存多一份副本，一旦Namenode发生故障，用来代替它，成为新的Namenode，使集群重新运行  它作用：  (1) 监控HDFS状态的辅助后台程序  (2) 每个集群只有一个Secondary Namenode进程  (3) 与Namenode进行通信，定期保存HDFS元数据快照  (4) Namenode故障时备用 |
| **JobTracker** | 处理任务跟踪进程，起到对MapReduce总控制和调度的作用。  它的特点：  (1) 是处理作业(用户任务)的后台程序，每个集群只有一个JobTracker进程，位于Master节点。  (2) 决定哪些文件参与处理，然后切割task并分配节点。  (3) 监控task，重启失败的task(在不同的节点)。 |
| **TaskTracker** | 任务跟踪进程，它的特点：  (1) 位于slave节点上，与DataNode进程结合(代码与数据一起的原则)  (2) 管理各自节点上的task(task由JobTracker分配)  (3) 每个节点只有一个TaskTracker进程，但是一个TskTracker可以启动多个JVM(java虚拟机)，用于并行执行map或reduce任务。  (4) 与JobTracker通信交互 |
| **DataNode** | 数据节点进程，它特点：  (1) 每个集群有很多个服务器节点，但是每台从服务器上只运行一个DataNode进程  (2) 负责把HDFS数据块读写到本地文件系统 |

Master进程包括：Namenode、Secondary Namenode、JobTracker

Slave进程包括TaskTracker、DataNode

集群时通常是Namenode、JobTracker这两个进程同时运行在同一台服务器，Secondary Namenode在另一台服务器运行。其他每个节点服务器都分别运行TaskTracker、DataNode两个进程。

# 3 hadoop安装

安装要求，至少要有三个服务器，一个作为master，另外两个作为slave

如果有空闲服务器，安装ESXi虚拟机系统(安装前不需要window或linux系统)，在ESXi虚拟机系统允许多少个虚拟机取决于服务器硬件配置。通过ESXi client客户端管理ESXi主机。

hadoop下载地址：<https://mirrors.tuna.tsinghua.edu.cn/apache/hadoop/common/>

## 3.1 逐个安装hadoop

### (1) 修改主机名

**vim /etc/hostname**

修改主机名称为VS01，注意这个名称在hadoop集群里是唯一的，不能相同。

注：hadoop集群中每个节点的名称都不一样

### (2) 把主机名映射到ip

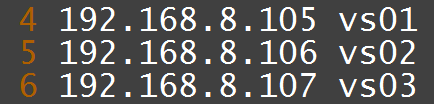
**vim /etc/hosts**

添加内容：

|  |
| --- |
| 192.168.8.105 vs01  192.168.8.106 vs02  192.168.8.107 vs03 |

当很多台主机集群时，也要把其他主机名和ip映射添加进来

修改如下图所示：



重启主机：**sudo reboot**

注：hadoop集群中每个节点的/etc/hosts内容是一样的

### (3) 创建hadoop运行的帐号

可以使用已存在用户，例如用户viosn，不建议hadoop在root权限下执行。

新建用户组和用户的名称都为hadoop

添加用户组：**groupadd hadoop**

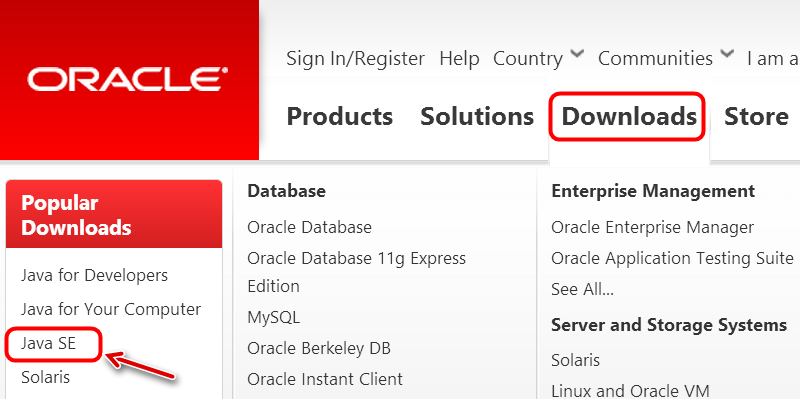
添加新用户：**useradd -s /bin/bash -d /home/vison -m vison -g hadoop**

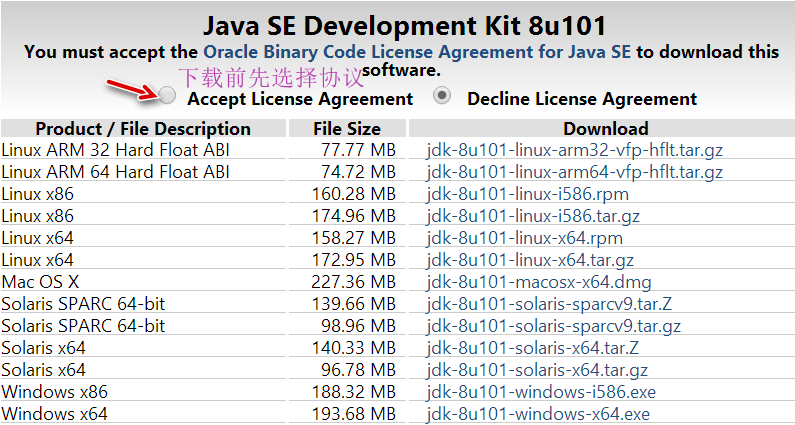
修改用户密码：**passwd vison**

注：hadoop集群中每个节点的帐号名称可以是一样

### (4) 安装JDK工具包

打开oracle官网[www.oracle.com](http://www.oracle.com)，选择java SE项，选择适合系统的版本下载，如下图所示：





安装步骤：

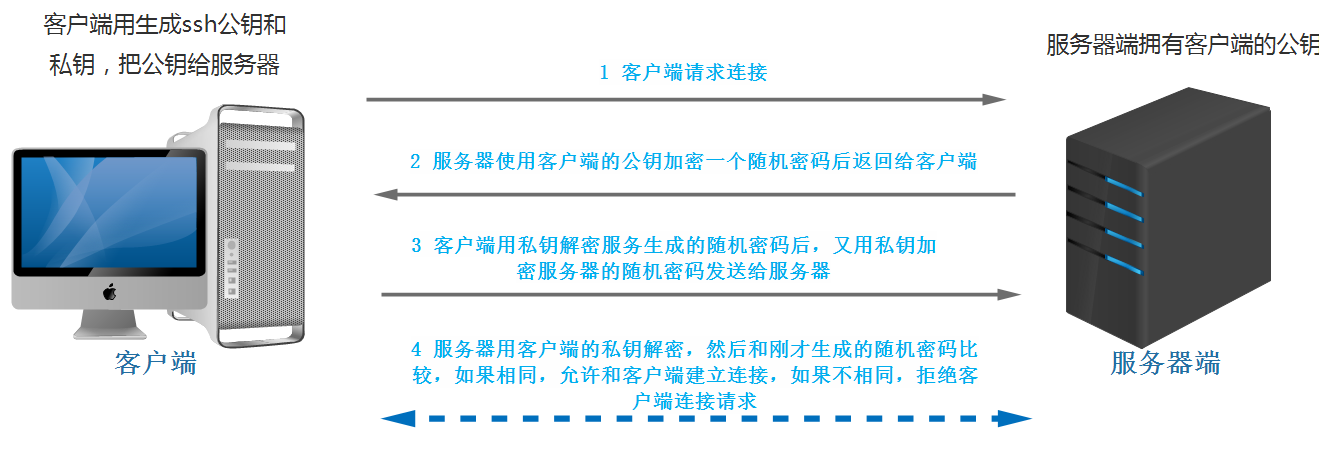
|  |  |
| --- | --- |
| 切换到hadoop运行的帐号 | su vison  cd ~/src |
| 下载JDK到linux | wget wget http://download.oracle.com/otn-pub/java/jdk/8u101-b13/jdk-8u101-linux-x64.tar.gz?AuthParam=1476349757\_cbde251d5f82c82827ef79cd3147bfd2 |
| 解压 | tar zxvf jdk-8u101-linux-x64.tar.gz\?AuthParam\=1476349757\_cbde251d5f82c82827ef79cd3147bfd2 |
| 把解压的文件夹移动安装目录下 | mv jdk1.8.0\_101 /home/vison/work |

注：hadoop集群中每个节点的JDK安装路径都一样

### (5) 设置ssh免密码登录

ssh的RSA不对称加密解密算法的特点：公钥机密的内容只有私钥才能解密，反过来私钥加密的内容只有公钥才能解密，公钥和私钥各自不能解密本身加密的内容，不能互相推导出对方的密钥，公钥可以向所有人公开，私钥为个人保管，

免密码登录原理如下图所示：



一般ssh远程登录时都需要手动输入密码，服务器通过ssh认证后才能允许免密码连接，只有设置好免密码登录功能才能完全实现hadoop的集群。

|  |  |
| --- | --- |
| 切换到hadoop运行的帐号 | su vison  cd ~ |
| 生成ssh私钥和公钥 | ssh-keygen -t rsa  注：生成的公私钥过程中一路按确认键默认即可，公私钥存放位置在生成过程中有说明，id\_rsa(私钥) id\_rsa.pub(公钥) |
| 把公钥复制为已认证的公钥keys里 | cd .ssh  cp id\_rsa.pub authorized\_keys  注意authorized\_keys这个文件的权限问题，不能让所有者之外的用户对authorized\_keys文件有写权限，否则会造成设置免密码登录无效。 |
| 合并所有节点的authorized\_keys文件 | 把所有节点的公钥并成一个大家共有公钥组群文件，然后替换每一个节点下的authorized\_keys  在vs02和vs03节点分别查看公钥内容：  cat ./authorized\_keys  在vs01节点操作：合并文件  vim. /authorized\_keys  把vs02和vs03内容复制进来，合并后文件给节点vs02和vs03  scp ./authorized\_keys vs02:/home/vison/.ssh  scp ./authorized\_keys vs03:/home/vison/.ssh |
| 验证是否完成免密码登录 | 各节点之间都要相互验证一下，因为第一次验证要手动允许 |

注：hadoop集群中每个节点的authorized\_keys文件都是一样的，所有的节点的公钥都存放在里面

### (6) 安装hadoop1.x版本

安装步骤：

|  |  |
| --- | --- |
| 切换到hadoop运行的帐号 | su vison  cd ~/src |
| 下载源码 | wget https://mirrors.tuna.tsinghua.edu.cn/apache/hadoop/common/hadoop-1.2.1/hadoop-1.2.1-bin.tar.gz |
| 解压 | tar zxvf hadoop-1.2.1-bin.tar.gz |
| 移动到安装目录 | mv hadoop-1.2.1 /home/vison/work |

注：hadoop集群中每个节点的hadoop安装路径都一样

hadoop配置文件都存放在安装目录下的conf文件夹下

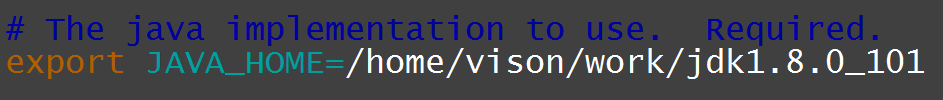
### (7) 修改hadoop-env.sh配置文件

**vim /home/vison/work/hadoop-1.2.1/conf/hadoop-env.sh**

指定JAVA\_HOME的所在的jdk路径

|  |
| --- |
| export JAVA\_HOME=/home/vison/work/jdk1.8.0\_101 |

修改结果如下图所示：



注：hadoop集群中每个节点的hadoop-env.sh内容都一样

### (8) 修改core-site.xml配置文件

指定本机地址(主机名)和端号，并且指定hadoop的临时路径，如果不指定hadoop临时路径，hadoop会把数据存放在linux系统tmp临时目录，重启系统后tmp内容清空，也就是hadoop文件被删除了。

**vim /home/vison/work/hadoop-1.2.1/conf/core-site.xml**

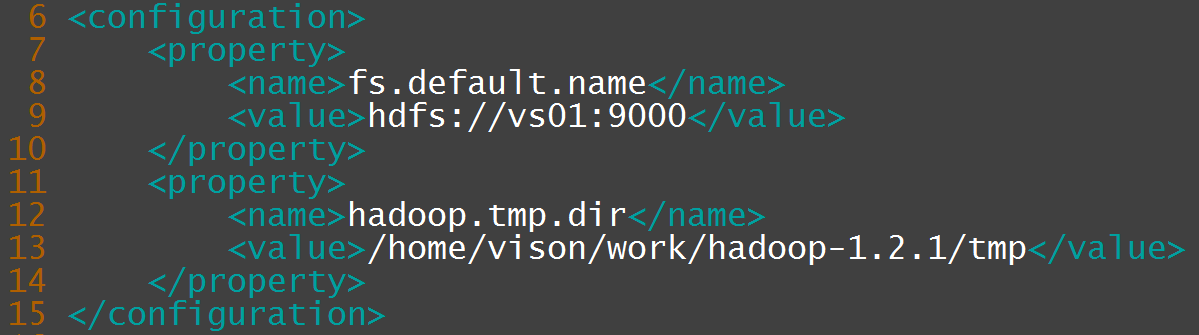
在<configuration>字段里添加下面内容：

|  |
| --- |
| <!-- 指定节点地址和端号 -->  <property>  <name>fs.default.name</name>  <value>hdfs://vs01:9000</value>  </property>  <!-- 指定临时文件存放位置-->  <property>  <name>hadoop.tmp.dir</name>  <value>/home/vison/work/hadoop-1.2.1/tmp</value>  </property> |

新建一个存放临时数据的文件夹：

**mkdir /home/vison/work/hadoop-1.2.1/tmp**

配置结果如下图：



注：hadoop集群中每个节点的core-site.xml内容都不一样，区别在指定各自的节点的地址

### (9) 修改hdfs-site.xml配置文件

指定dfs文件存放在linux的位置和dfs文件复制因子，也就是一个文件在分布式文件系统中复制多少份相同的文件，复制的文件分布在不同的节点上。

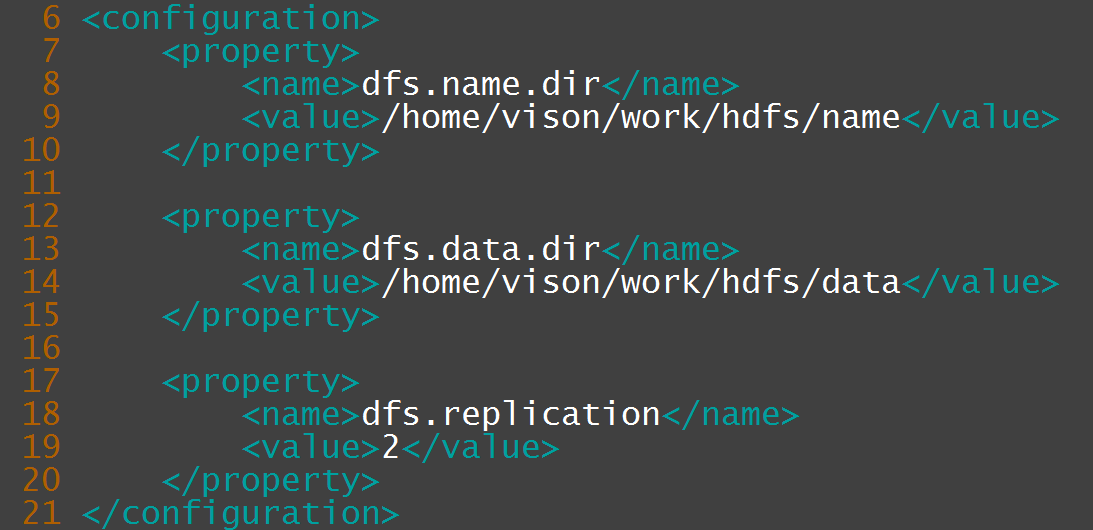
**vim /home/vison/work/hadoop-1.2.1/conf/hdfs-site.xml**

在<configuration>字段里添加下面内容：

|  |
| --- |
| <!-- 设置hdfs名称 -->  <property>  <name>dfs.name.dir</name>  <value>/home/vison/work/hdfs/name</value>  </property>  <!-- 设置hdfs数据存放路径 -->  <property>  <name>dfs.data.dir</name>  <value>/home/vison/work/hdfs/data</value>  </property>  <!-- 设置文件复制因子 -->  <property>  <name>dfs.replication</name>  <value>2</value>  </property> |

复制因子越大，文件越安全，但是冗余文件越多，占用硬盘空间越大，根据实际场景选取一个平衡点在确定复制因子的值，如果设置为1表示不复制。

修改结果如下图：



注：hadoop集群中每个节点的hdfs-site.xml内容都一样

### (10) 修改mapred-site.xml配置文件

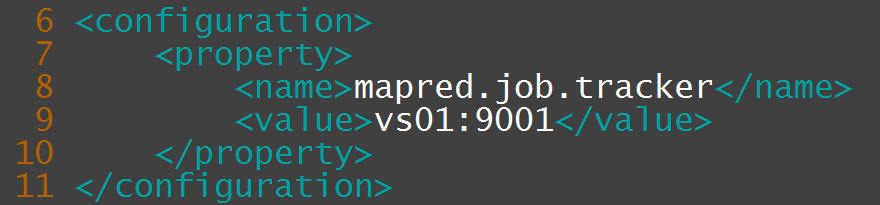
指定JobTracker进程监听地址(主机名:9001)

**vim /home/vison/work/hadoop-1.2.1/conf/mapred-site.xml**

在<configuration>字段里添加下面内容：

|  |
| --- |
| <property>  <name>mapred.job.tracker</name>  <value>vs01:9001</value>  </property> |

修改结果如下图所示：



注：hadoop集群中每个节点的mapred-site.xml内容都一样

### (11) 修改masters和slaves配置文件

主节点只有一个，数据节点有多个，当增加数据节点时，只需在主节点上修改slaves文件，其他数据节点不需要修改，因为整集群启动是在主节点，启动时主节点读取masters和slaves文件就知道哪些可用的数据节点。

修改masters文件：

**vim /home/vison/work/hadoop-1.2.1/conf/masters**

添加内容：

|  |
| --- |
| vs01 |

修改结果如下图所示：



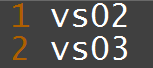
修改slaves文件：

**vim /home/vison/work/hadoop-1.2.1/conf/slaves**

添加内容：

|  |
| --- |
| vs02  vs03 |

修改结果如下图所示：



### (12) 修改环境变量

**vim /etc/profile**

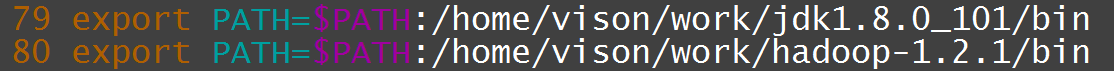
添加内容：

|  |
| --- |
| #JDK工具命令  export PATH=$PATH:/home/vison/work/jdk1.8.0\_101/bin  #hadoop命令  export PATH=$PATH: /home/vison/work/hadoop-1.2.1/bin |

使环境变量生效：

**source /etc/profile**

修改结果如下图所示：



注：hadoop集群中每个节点的profile内容都一样

### (13) 验证hadoop是否安装成功

注意：下面步骤必须在hadoop运行的帐号下操作

**① 初始化hadoop名称节点**

在名称节点上格式化：

**hadoop namenode -format**

初始化一系列工作，包括日志、文件系统等，如果没有出现warning或error，说明格式化成功，一般warning都是配置有误造成的，格式化结果如下图所示：

注：在hadoop format过程中 要注意不要频繁地reformat  namnode的ID信息。format过程中选择N就是了，否则向hadoop put文件时会出现**could** **only** **be** **replicated** **to** > **0** **nodes,** **instead** **of** 1错误，错误解决办法在最后。

**② 启动hadoop集群**

在namenode节点下执行启动hadoop集群：

**start-all.sh**

start-all.sh意思是启动所有hadoop集群(包括主节点和数据节点)，需要停止整个hadoop集群用stop-all.sh

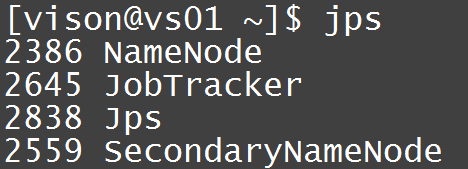
③ 检测hadoop守护启动进程情况

使用JDK工具命令查看hadoop集群进程：

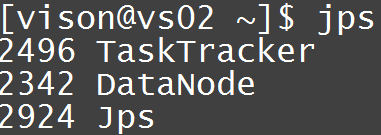
**jps**

jps类似linux的ps命令

主节点启动进程如下图所示，出现Namenode、SecondaryNamenode、JobTracker进程，说明主节点启动成功



数据节点启动进程如下图所示，出现DataNode、TaskTracker进程，说明数据节点也启动成功。

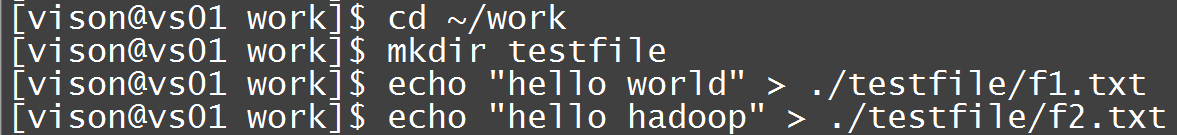


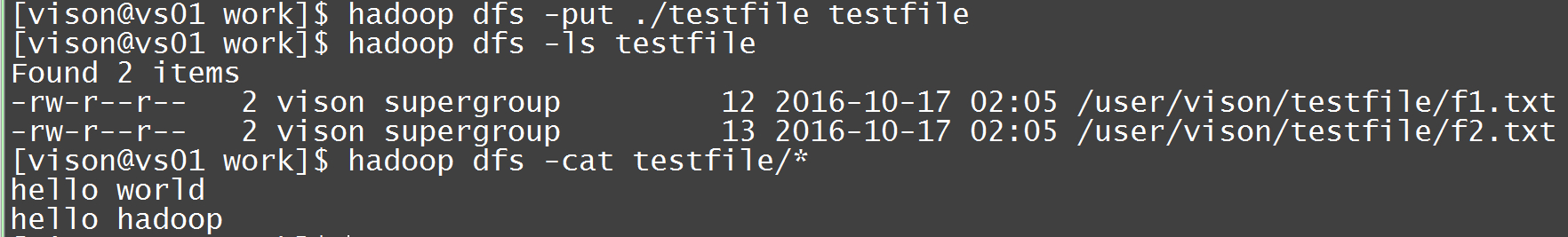
**(14) 对hadoop集群做测试**

把linux下本地的两个文件发送给hadoop

|  |
| --- |
| // 新建两个测试文件  cd ~/work  mkdir testfile  echo "hello world" > ./testfile/f1.txt  echo "hello hadoop" > ./testfile/f2.txt  // 把两个文件传输到hadoop的testfile目录下  hadoop dfs -put ./testfile testfile  // 查看hadoop文件目录  hadoop dfs -ls testfile  // 查看刚上传给hadoop文件内容  hadoop dfs -cat testfile/\* |

操作结果：





在网页查看Namenode信息

<http://192.168.8.105:50070/dfshealth.jsp>

在网页查看JobTracker信息

<http://192.168.8.105:50030/jobtracker.jsp>

总结常出现错误如下:

* 系统或hdfs是否有足够空间
* datanode数是否正常
* 是否在safemode
* 防火墙是否关闭
* 关闭hadoop、格式化、重启hadoop

**常见错误1**

在hadoop上put数据时出现

WARN hdfs.DFSClient: Error Recovery for null bad datanode[0] nodes == null

**could** **only** **be** **replicated** **to** > **0** **nodes,** **instead** **of** 1的错误提示

原因：是由于多次format namenode节点导致namdenode的namespaceID与 datanode的namespaceID不一致，从而导致namenode和datanode的断连。

**解决办法：**

* 停止hadoop：stop-all.sh
* 把所有节点dfs.name.dir路径和dfs.data.dir路径下的文件删除(退回到Namenode格式化前状态)
* 格式化hadoop：**hadoop namenode -format**
* 需要先启动namenode，再启动datanode，然后启动jobtracker、tasktracker、secondarynamenode。

hadoop-daemon.sh start namenode

hadoop-daemon.sh start datanode

hadoop-daemon.sh start jobtracker

hadoop-daemon.sh start tasktracker

hadoop-daemon.sh start secondarynamenode

* 再次停止hadoop：stop-all.sh
* 启动hadoop：start-all.sh

## 3.2 实际集群安装hadoop

安装大集群按需要awk写脚本，生成脚本命令模式

cat ./hosts | awk '{print "scp -rp 本机文件路径 用户名@"$1":远程主机文件路径"}' >> run.sh

说明：scp -r表示复制包括文件子目录内容，-p表示保持文件的拥有者和权限

要使脚本能够执行成功，必须先完成各节点免密码配置。

**(1) 配置host文件**

host文件内容是主机名和IP地址的映射，假如每增加一个节点，每个节点的host文件都要把新增节点的主机名和IP地址添加进来，如果有上百台服务器，每个节点都是手工去配置是一个很大工作量。

通常使用DNS服务器来解决，DNS是把域名翻译成IP地址的服务器，在DNS服务器记录每个节点的主机名和IP地址映射，通过主机名称就可以解析出IP地址来，当增加一个节点时，只需要把新增节点主机名和IP地址记录在DNS服务器上，这样集群上每个节点都不需要修改host文件内容，不再需要维护host文件，只需维护DNS服务器。

在linux系统安装bing软件后就可以为DNS服务器，然后在集群每个节点修改resolv.conf配置文件

**vim /etc/resolv.conf**

添加内容：

|  |
| --- |
| nameserver 192.168.8.8 # 本地DNS服务器IP地址  #可以指向其他DNS服务器 |

测试是否能成功解析主机名命令：nslookup

然后输入本机主机名，如果能解析出IP说明DNS服务器解析正常

(2) 建立hadoop运行帐号

awk写脚本

**(3) 配置ssh免密码连入**

配置ssh免密码时需要获取每个节点的公钥文件内容，把各节点内容合并，然后再发送会给每个节点，当每增加一个节点时候，每个节点又重复一遍这样的操作，手工操作的话是非常繁琐的。

通常解决办法是NFS或写脚本替换。

(4) 下载并解压hadoop安装包

只需在一台节点下载，其他节点通过复制完成

(5) 配置namenode，修改site配置文件

awk写脚本

(6) 配置hadoop-env.sh

awk写脚本

**(7) 配置masters和slaves文件**

并不是每台节点都需要配置masters和slaves文件，只需要修改Namenode节点的主机上的masters和slaves文件，不需要每个节点更改。例如增加一个节点，直接在Namenode节点的主机上slaves增加即可

也可以用awk写脚本

**(8) 向各节点复制hadoop程序**

通过awk工具生成脚本文件，执行脚本文件自动完成向个各个节点复制到指定目录，首先把各主机名列在文件名为hosts上，hosts内容是每行一个主机名称，执行命令：

**cat ./hosts | awk '{print "scp -rp /home/viosn/work/hadoop-1.2.1 vison@"$1":/home/viosn/work"}' >> copy.sh**

通过这条命令生成一个脚本，这个脚本的功能是通过ssh把hadoop软件逐个向各节点推送，整个推送过程是自动的，无序人工干预，最后执行copy.sh

脚本文件即可。其他配置文件也可以通过这个方法来完成。

(9) 格式化namenode

(10) 启动hadoop

(11) 用jps检验各后台进程是否启动成功