**大数据**

大数据介绍

大数据是做什么的

大数据

随着计算机技术的发展，互联网的普及，信息的积累已经到了一个非常庞大的地步，信息的增长也在不断的加快，随着互联网、物联网建设的加快，信息更是爆炸性的增长，收集、检索、统计这些信息越发困难，必须使用新的技术来解决这些问题

什么是大数据

大数据的定义

大数据由巨型数据集组成，这些数据集大小常超出人类可接受时间下的收集、调用、管理和能力。

大数据无法在一定时间范围内用常规软件工具进行捕捉、管理和处理的数据集合，是需要新处理模式才能具有更强的决策力、洞察发现力和流程优化能力的海量增长、高增长率和多样化的信息资产。

大数据是指无法在一定时间内用常规工具对其内容进行抓取、管理和处理的数据集合。大数据技术，是指从各种各样的数据中，快速获取有价值的信息的能力。适用于大数据的技术，包括大规模并行处理数据库，数据挖掘网，分布式文件系统，分布式数据库，云计算平台，互联网，和可扩展的存储系统。

大数据能做什么

把数据集合并后进行分析可得出许多额外的信息和数据关系性，可用来察觉商业趋势、判定研究质量、避免疾病扩散、打击犯罪或测定即时交通路况等；这样的用途正是大数据盛行的原因。

企业组成利用相关数据和分析可以帮助它们降低成本、提高效率、开发新产品、做出更明智的业务决策等等

大数据的特性



大数据的5V特性是什么

(V)olume（大体量）

即可从数百TB到数十数百PV、甚至EB的规模

(V)ariety（多样性）

即大数据包括各种格式和形态的数据

(V)elocity（时效性）

即很多大数据需要在一定的时间限度下得到及时的处理

(V)eracity（准确性）

即处理的结果哟保证一定的准确性

(V)alue（大价值）

即大数据包含很多深度的价值，大数据分析挖掘和利用将带来巨大的商业价值

大数据与hadoop

hadoop是什么

hadoop是一种分析和处理海量数据的软件平台

hadoop是一款开源软件，使用java开发

hadoop可以提供一个分布式基础架构

hadoop特点

高可用性、高扩展性、高效性、高容错性、低成本

hadoop历史起源

hadoop起源

2003年开始google陆续发表了几篇论文

GFS,MapReduce,BigTable

GFS是一个可扩展的分布式文件系统，用于大型的、分布式的、对大量数据进行访问的应用。它运行于廉价的普通硬件上，提供容错功能

Mapreduce是针对分布式并行计算的一套编程模型。是由map和reduce组成，map是影射，把指令分发多多个worker上去，reduce是规约，把map的worker计算出来的结果合并

Bigtable存储结构化数据。是建立在GFS,Scheduler,lock service和mapreduce之上的。每个table都是一个多维的稀疏图

这三个技术被称为googole的三驾马车。虽然google没有公布这三个产品的源码，但是发布了这三个产品的详细设计论文。yahoo资助的hadoop按照这三篇论文的开源java实现，不过在性能上hadoop比goole要差很多

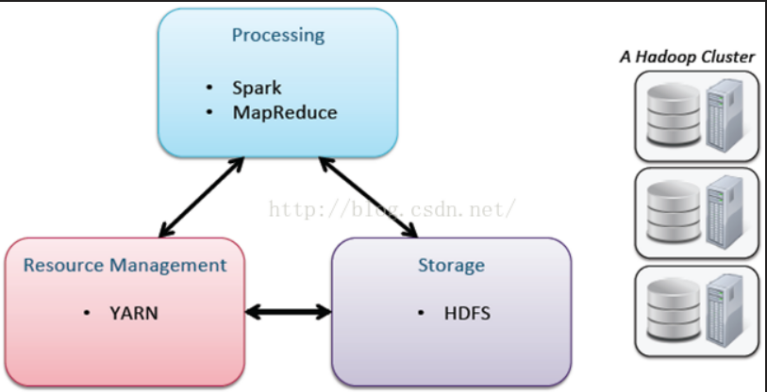
GFS--->HDFS

Mapreduce--->mapreduce

bigtable--->hbase

hadoop组件

Hadoop核心组件

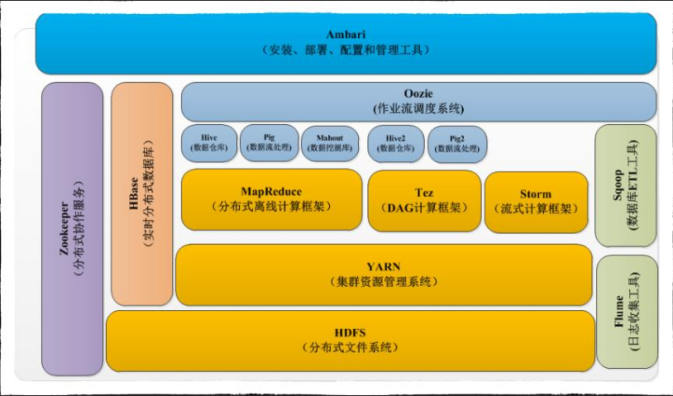


HDFS：分布式文件系统

Mapreduce：分布式计算框架

yarn：集群资源管理系统

hadoop生态环境



hadoop常用组件

HDFS（hadoop分布式文件系统）

Mapreduce（分布式计算框架）

Zookeeper（分布式协作服务）

Hbase（分布式列存数据库）

Hive（基于hadoop的数据仓库）

Sqoop（数据同步工具）

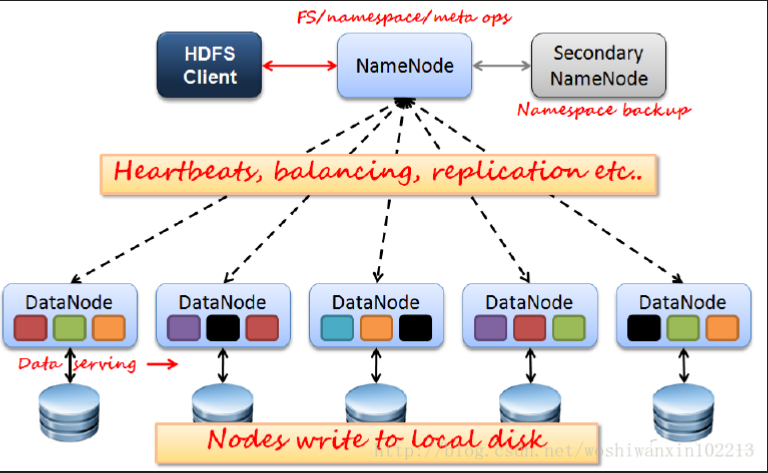
Pig（基于hadoop的数据流系统）

Mahout（数据挖掘算法库）

Flume（日志收集工具）

hadoop核心组件

HDFS结构（分布式文件系统）



HDFS角色及概念

是hadoop体系中数据存储管理的基础。它是一个高容错的系统，用于在低成本的通用硬件上运行。

角色和概念

client

namenode

secondarynode

datanode

namenode

master节点，管理HDFS的名称空间和数据块映射信息，配置副本策略，处理所有客户端请求，把数据存储到datanode

secondaty namenode

定期合并fsimage（数据记录日志）和fsedits（数据变更日志），推送给namenode。紧急情况下，可辅助恢复namenode。但secondary namenode并非namenode的热备

datanode

数据存储节点，存储实际的数据，汇报存储信息给namenode

client

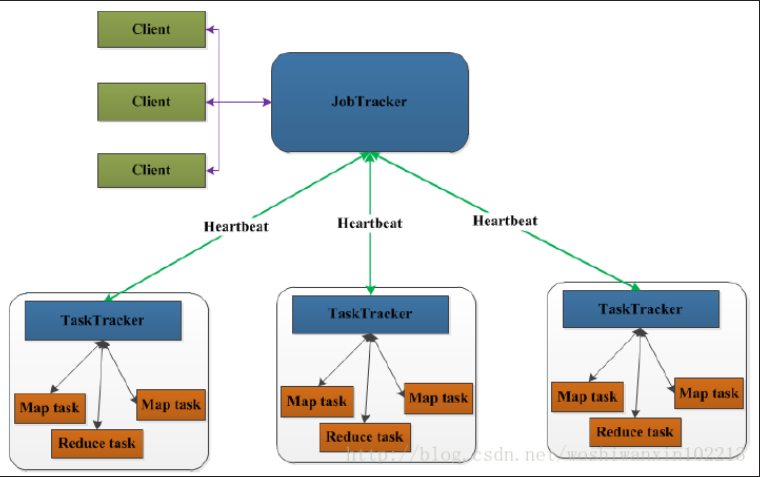
切分文件。访问HDFS。与NameNode交互，获取文件位置信息。与Datanode交互，读取和写入数据。

block

每块默认64MB大小，每块可以多个副本

client将数据切片发送给namenode，namenode存储到datenode节点上，并写入日志fsimage和fsedits，secondary namenode定期将连个日志进行整理，合并后发送给namenode

Mapreduce结构（分布式计算框架）



mapreduce角色及概念

源自于google的mapreduce论文，java实现的分布式计算框架

角色和概念

jobtracker

tasktracker

map task

reducer task

jobtracker

master节点。只有一个

管理所有作业

作业/任务的监控、错误处理等

将任务分解成一系列任务，并分派给tasktracker

tasktracker

slave节点，一般是多台

运行map task和reduce task

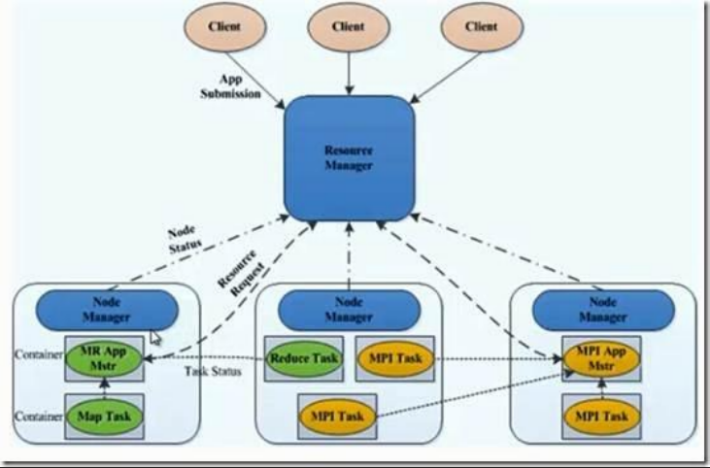
并与jobtracker交互，汇报任务状态

map task：解析每条数据记录，传递给用户编写的map()，并执行，将输出结果写入本地磁盘（如果为map-only作业，直接写入HDFS）

reduce task：从map task的执行结果中，远程读取输入数据，对数据进行排序，将数据按照分组传递给用户编写的reduce函数执行

客户端发送请求给jobtracker，jobtracker分配任务给tasktracker，tasktracker启动map task和reduce task执行，执行完的结果返回给jobtracker，jobtracker再3返回给客户端

yarn结构（集群资源管理）



yarn角色及概念

yarn是hadoop的一个通用的资源管理系统

yarn角色

resourcemanager

nodemanager

applicationmaster

container

client

resourcemanager

处理客户端请求

启动 /监控applicationmaster

监控nodemanager

资源分配与调度

nodemanager

单个节点上的资源管理

处理来自resourcemanager的命令

处理来自applicationmaster的命令

container

对任务运行环境的抽象，封装了CPU、内存等

多维资源以及环境变量、启动命令等任务运行相关的信息资源分配与调度

applicationmaster

数据切分

为应用程序申请资源，并分配给内部任务

任务的监控与报错

client

用户与yarn交互的客户端程序

提交应用程序、监控应用程序状态，杀死应用程序等

yarn的核心思想

将jobtracker和tasktacker进行分离，它由下面几大构成组件：

resourcemanager一个全局的资源管理器

nodemanager每个节点（RM）代理

applicationmaster表示每个应用

每一个applicationmaster有多个container在nodemanager上运行

客户端发送请求给resourcemanger，resourcemanger将客户请求发送给nodemanager并启动applicationmaster程序，nodemanager处理来自resourcemanager的命令，处理完成后逐层返回

hadoop安装配置

hadoop的部署模式有三种

单机

伪分布式

完全分布式

hadoop单机模式安装配置

hadoop的单机模式安装非常简单

1. 获取软件

http://hadoop.apache.org

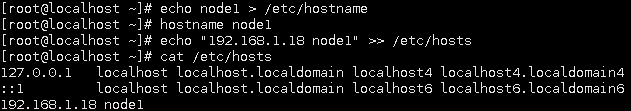
1. 安装配置java环境，安装jps工具

安装openjdk和openjdk-devel

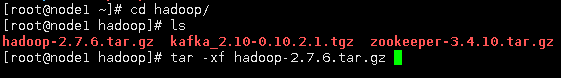
1. 设置环境变量，启动运行

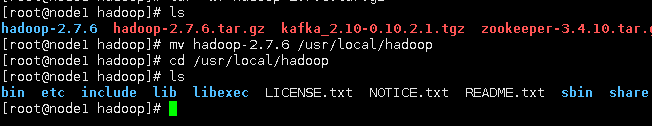
hadoop-env.sh

JAVA\_HOME=””

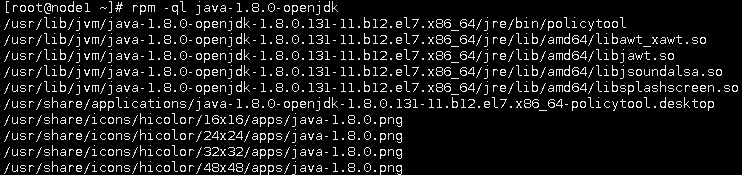




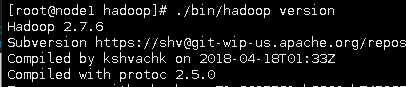






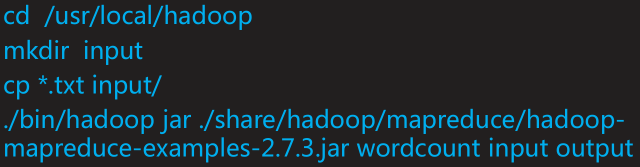


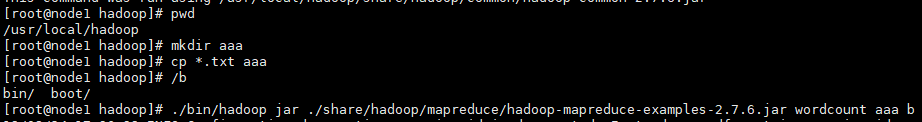




hadoop的单机模式安装非常简单，只需要配置好环境变量即可运行，这个模式一般用来学习和测试hadoop的功能

测试---统计词频





调用/bin/hadoop文件执行jar脚本，脚本是share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.6.jar，用来统计aaa目录里词的数量，放到b里面，b必须是不存在的目录，否则会报错

hadoop伪分布式

伪分布式

伪分布式的安装与完全分布式类似，但区别是所有角色安装在一台机器上，使用本地磁盘，一般生产环境都会使用完全分布式，伪分布式一般用来学习和测试方面的功能

伪分布式的配置和完全分布式配置类似

hadoop-env.sh

JAVA\_HOME

HADOOP\_CONF\_DIR

xml文件配置格式

<property>

<name>关键字</name>

<value>变量值</value>

<description> 描述 </description>

</property>

core-site.xml

关键配置

fs.defaultFS

hdfs://localhost:9000

常用配置

hadoop.tmp.dir

hdfs-site.xml

dfs.replication

dfs.namenode.name.dir

dfs.datanode.data.dir

dfs.namenode.http-address

dfs.namenode.secondary.http-address

dfs.webhdfs.enabled

mapreduce-site.xml

必选

mapreduce.framework.name

yarn

yarn-site.xml

yarn.resourcemanager.hostname

resourcemanager主机名

yarn.nodemanager.aux-services

mapreduce\_shuffle

HDFS分布式文件系统

什么是分布式文件系统

分布式文件系统是指文件系统管理的物理存储资源不一定直接连接在本地节点上，而是通过计算机网络与节点相连

分布式文件系统的设计基于客户机/服务器模式

一个典型的网络可能报错多个供多用户访问的服务器

对等特性允许一些系统扮演客户机和服务器的双重角色

分布式文件系统的特点

分布式文件系统可以有效解决数据的存储和管理难题

将固定于某个地点的某个文件系统，扩展到任意多个地点/多个文件系统

众多的节点组成一个文件系统网络

每个节点可以分布在不同的地点，通过网络进行节点间的通信和数据传输

人们在使用分布式文件系统时，无需关心数据是存储在哪个节点上，或者是从哪个节点上获取的，只需要像使用本地文件系统一样管理和存储文件系统中的数据

完全分布式

hadoop最大的优势就是分布式集群计算，所以在生产环境下都是搭建最后一种模式：完全分布式

技术准备

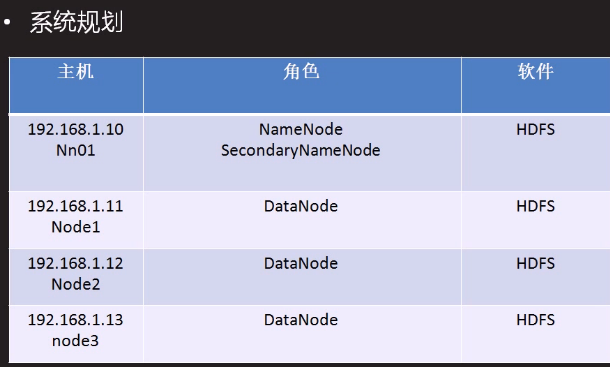
系统搭建

搭建

测试

上线使用

系统规划



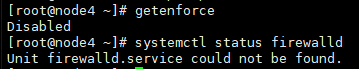
环境准备

新开启3台虚拟机，共四台

注意：只开启必要的服务，关闭其他无关的系统服务，系统最小化，服务最小化

注意：禁用selinux

注意：禁用firewalld



在所有主机上安装java环境和调试工具jps

注意：保证所有机器系统版本及java版本的一致性

注意：保证所有安装路径的一致性

安装java-1.8.0-openjdk-devel

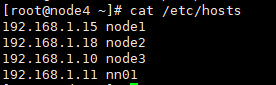


配置主机名和ip对应关系/etc/hosts

注意：所有主机的配置都要修改

注意：master要能ping通所有主机

注意：node要能ping通master



配置ssh信任关系（namenode）

注意：不能出现要求输入yes的情况，每台机器都要登录成功，包括本机

ssh\_config

StrictHostKeyChecking no

ssh-keygen -b 2048 -t rsa -N '' -f key

ssh-copy-id -i [root@ip.xx.xx.xx](mailto:root@ip.xx.xx.xx)

















HDFS完全分布式系统配置

环境配置文件hadoop-env.sh

核心配置文件core-site.xml

HDFS配置文件hdfs-site.xml

节点配置文件 slaves

slaves

node1

node2

node3





hadoop-env.sh

which java

readlink –f $(which java)

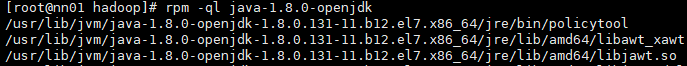
JAVA\_HOME=””

HADOOP\_CONF\_DIR=${HADOOP\_CONF\_DIR:-

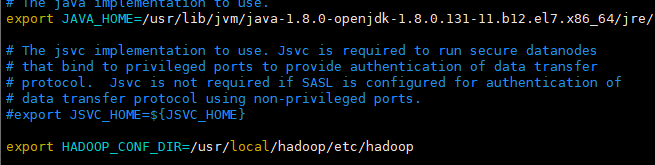
"/etc/hadoop"}

配置HADOOP\_CONF\_DIR路径

/usr/local/hadoop/etc/hadoop







core-site.xml

fs.defaultFS文件系统配置参数

hadoop.tmp.dir数据目录配置参数

<property>

<name>fs.defaultFS</name>

<value>hdfs://192.168.4.10:9000</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/var/hadoop</value>

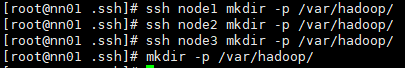
<description>A base for other temporary

directories.</description>

</property>



注意：所有机器上都要创建mkdir -p /var/hadoop



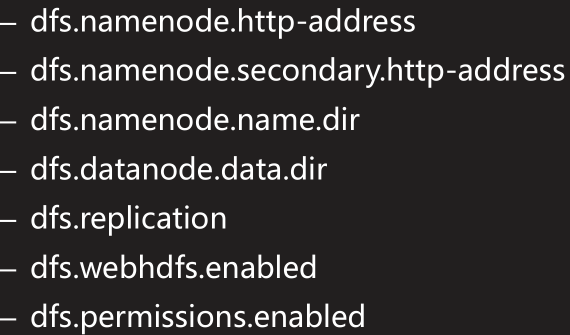
hdfs-site.xml

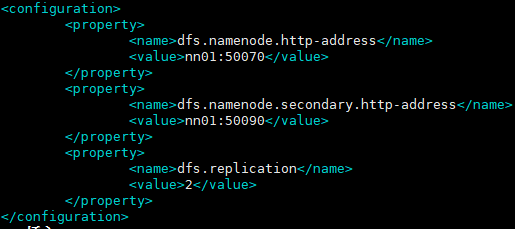
<property>

<name>dfs.replication</name>

<value>2</value>

</property>



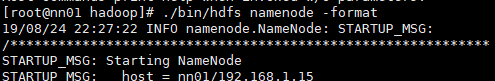


配置完成以后，把hadoop的文件夹拷贝到所有机器



在namenode上执行格式化操作

./bin/hdfs namenode -format





在没有报错的情况下启动集群

./sbin/start-dfs.sh



停止集群可以使用./sbin/stop-dfs.sh

验证

角色

启动后分别在namenode和datanode执行命令

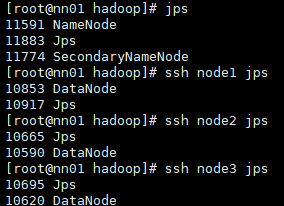
jps

成功的情况下应该可以看见

namenode

secondarynode

datanode



集群是否组建成功

./bin/hadoop dfsadmin -report

服务器启动日志

/usr/local/hadoop/logs