# 知识回顾

## 初识大数据

### 什么是大数据？

大数据就是海量数据的处理，用传统方式处理不了的数据。

### 大数据是怎么来的？

互联网带来了数据爆炸时代。

### 各种疑难问题的处理方式

分布式在某种意义上更加考验程序员的能力，加大了开发难度，其实本质是硬件的妥协。

高并发🡪分布式

高可用🡪分布式

大数据🡪分布式

### 大数据相关技术

Flume：数据的收集聚集加载

Hadoop.HDFS：数据的存储

MapReduce、Hive、SparkSql：数据的离线处理

Storm、SparkStreaming：数据的实时处理

Hbase：数据库

Sqoop：HDFS和关系型数据库桥梁

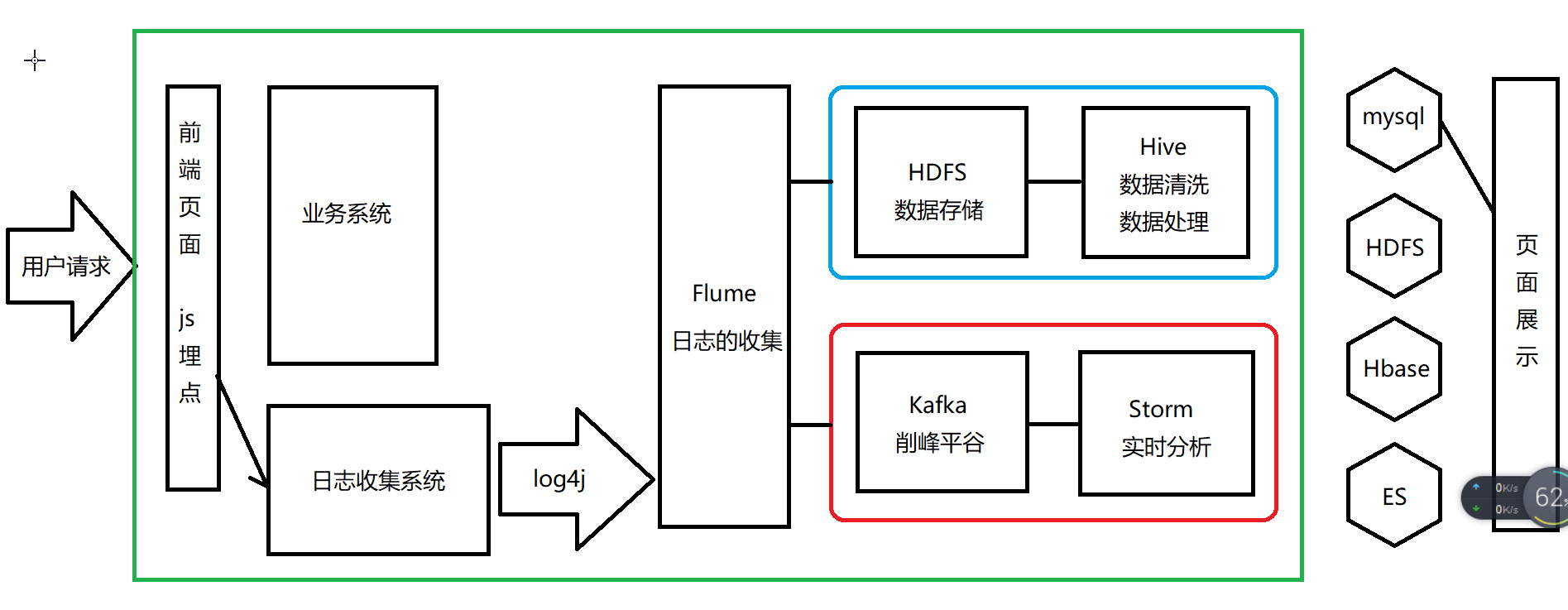
## 日志流量分析系统

### 项目需求

京淘的第八个子项目，主要用于处理分析京淘访客日志，以协助管理者提高决策能力。

### 项目设计

项目通用，可以对接任何有相应需求的项目



### 日志的捕获

JS埋点：将参数系数组织起来发送到日志收集系统。（传统后台埋点，影响后台维护和运行；script标签内发送影响前端逻辑，最后选择以img标签实现埋点信息的发送）

Log4j：收集日志，在打印同时，输出到flume中。

Flume：数据的收集聚集加载

### 离线分析

HDFS（Hadoop的一部分）：海量数据的存储

Hive（底层基于Hadoop）：海量离线数据的处理

### 实时分析

Storm：流式数据处理（速度最快）

Kafka：消息队列，削峰平谷

## 八个重要指标

PV:点击量

UV:独立访客数

VV:独立会话数

BR:跳出率

NewIP:新增独立IP

NewCust:新增独立访客

AvgTime:平均访问时长

AvgDeep:平均访问深度

## 系统搭建

独立的服务器，逻辑简单，用Servlet处理请求参数，进行字符串拼接，并通过log4j输出到控制台和flume中。

# Hadoop

## Hadoop概述

Hadoop是大数据领域中非常重要的基础技术，他是一个海量数据存储、处理系统，也是一个生态圈（HDFS，MapReduce，Hive，Hbase。。。。Spark底层也有Hadoop）

### 历史

Google：搜索引擎。收集互联网上的所有数据，存储数据，处理数据，提供给用户。

Google搜索引擎相关技术非常成熟，但是并没有开源，不过，在2004年先后发表了两篇论文：《Google File System》（GFS）、《MapReduce》阐述了Google如何将海量数据进行存储和处理。2006年发表了《BigTable》启发了无数的NoSql数据库。

Doug Cutting（狗哥）：Lucene、Nutch（搜索引擎）

Hadoop（Java开发）

HDFS（Hadoop distributed file system）：大数据存储

MapReduce：大数据处理

### 作用

Hadoop最初用作Nutch底层的海量数据存储和处理，后来人们发现他也非常适合大数据场景下的数据存储和处理，主要用作海量离线数据的存储和处理。

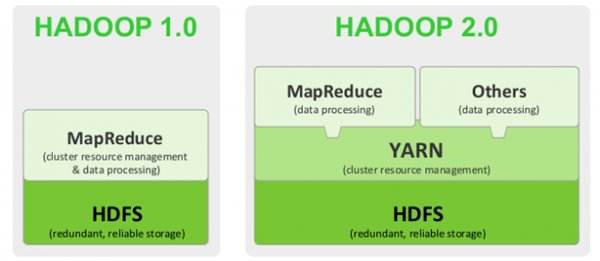
## Hadoop的安装

### Hadoop版本介绍

Hadoop1.0（Apache）最初版:HDFS、MapReduce

Hadoop2.0：HDFS、MapReduce、Yarn--------这个是我们要讲的！！！（2.7.1）

Hadoop3.0：2017年12月发布，目前还不是稳定版本。



### Hadoop 的安装有三种方式

单机模式：解压就能运行，但是只支持MapReduce的测试，不支持HDFS，不用。

伪分布式模式：单机通过多进程模拟集群方式安装，支持Hadoop所有功能。学习测试用

完全分布式模式：集群方式安装，支持高可用，最后一天进阶学习。

### Hadoop伪分布式安装

需要环境：

JDK，JAVA\_HOME，配置hosts，hostname，关闭防火墙，配置免密登录。

安装在hadoop01节点上。

1. 创建目录

mkdir hadoop

1. 上传安装包并解压

tar -xvf 安装包

1. 修改配置文件

**1.修改 hadoop-env.sh**

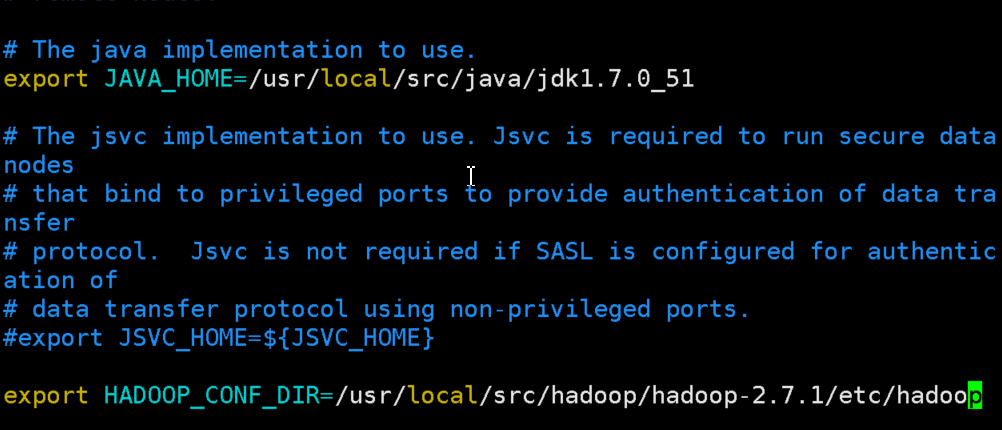
* 通过vim打开

vim [hadoop]/etc/hadoop/hadoop-env.sh

主要是修改java\_home的路径

在hadoop-env.sh的第25行,把export JAVA\_HOME=${JAVA\_HOME}修改成具体的路径

在33行，修改HADOOP\_CONF\_DIR为具体的路径



* 重新加载使修改生效

source hadoop-env.sh

**2.修改 core-site.xml**

* 通过vim打开

vim [hadoop]/etc/hadoop/core-site.xml

* 增加namenode配置、文件存储位置配置：粘贴高亮部分到<configuration>标签内

<configuration>

<property>

<!--用来指定hdfs的老大，namenode的地址-->

<name>fs.default.name</name>

<value>hdfs://hadoop01:9000</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>/usr/local/src/hadoop/hadoop-2.7.1/tmp</value>

</property>

</configuration>

**3.修改 hdfs-site.xml**

* 通过vim打开

vim [hadoop]/etc/hadoop/hdfs-site.xml

* 配置包括自身在内的备份副本数量：粘贴高亮部分到<configuration>标签内

<configuration>

<property>

<!--指定hdfs保存数据副本的数量，包括自己，默认为3-->

<!--伪分布式模式，此值必须为1-->

<name>dfs.replication</name>

<value>1</value>

</property>

<!--设置hdfs操作权限，false表示任何用户都可以在hdfs上操作文件-->

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

</configuration>

**4.修改 mapred-site.xml**

说明：在/etc/hadoop的目录下，只有一个mapred-site.xml.template文件，复制一个。

cp mapred-site.xml.template mapred-site.xml

* 通过vim打开

vim [hadoop]/etc/hadoop/mapred-site.xml

* 配置mapreduce运行在yarn上：粘贴高亮部分到<configuration>标签内

<configuration>

<property>

<!--指定mapreduce运行在yarn上-->

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>

**5.修改 yarn-site.xml**

* 通过vim打开

vim [hadoop]/etc/hadoop/yarn-site.xml

* 配置：粘贴高亮部分到<configuration>标签内

<configuration>

<property>

<!--指定yarn的老大resourcemanager的地址-->

<name>yarn.resourcemanager.hostname</name>

<value>hadoop01</value>

</property>

<property>

<!--NodeManager获取数据的方式-->

<name>yarn.nodemanager.aux-services</name>

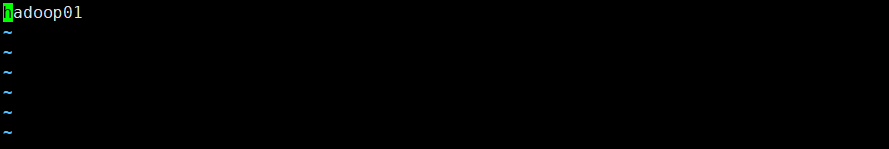
<value>mapreduce\_shuffle</value>

</property>

</configuration>

1. **修改 slaves**

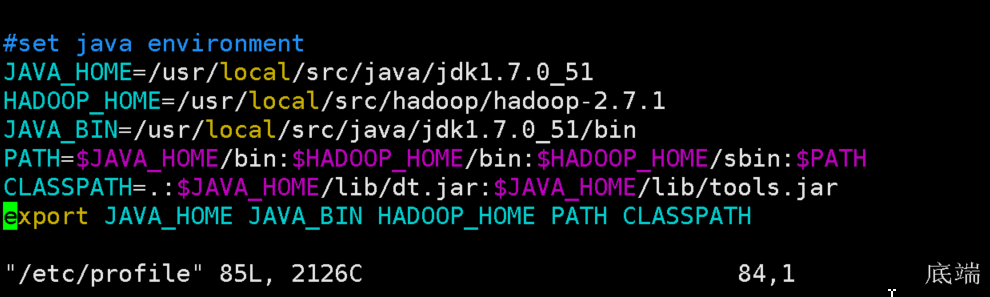
vim slaves



**7.配置hadoop的环境变量**

vim /etc/profile

最终结果如下图：



* 重新加载profile使配置生效

source /etc/profile

* 环境变量配置完成，测试环境变量是否生效

echo $HADOOP\_HOME

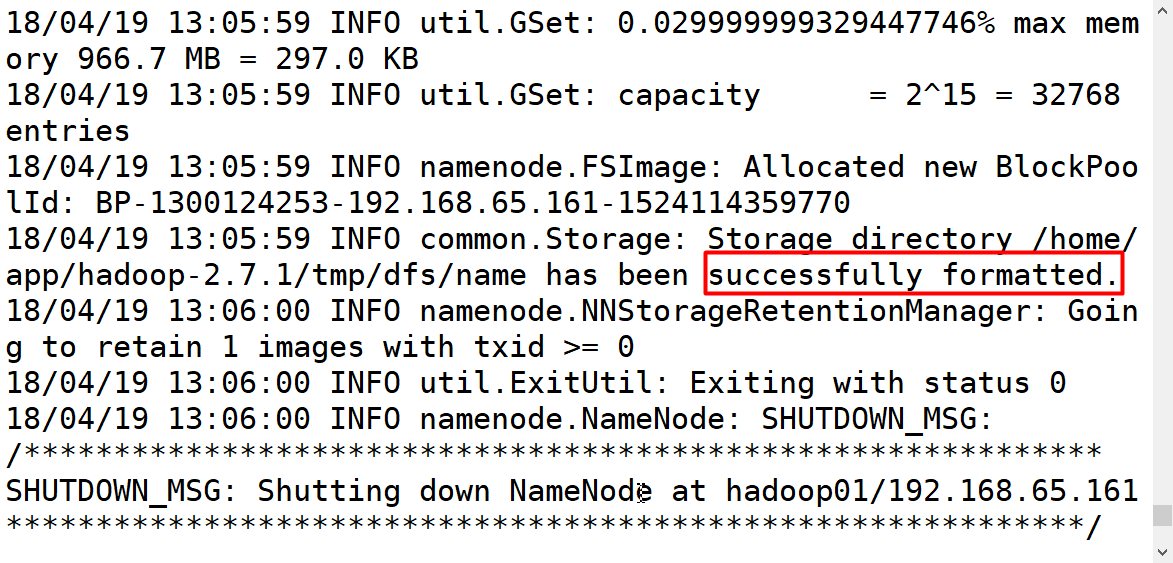
1. **创建tmp文件夹**

cd /usr/local/src/hadoop/hadoop-2.7.1

mkdir tmp

1. 初始化

bin/hdfs namenode –format



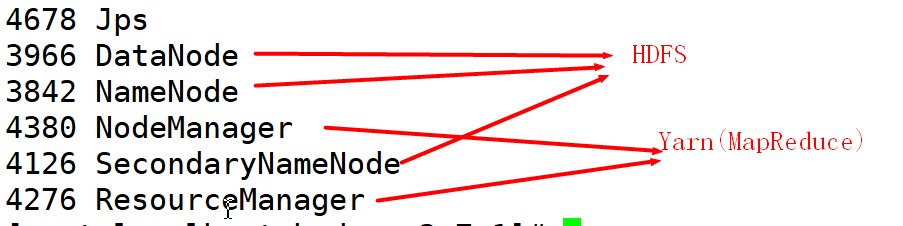
1. 启动

start-all.sh

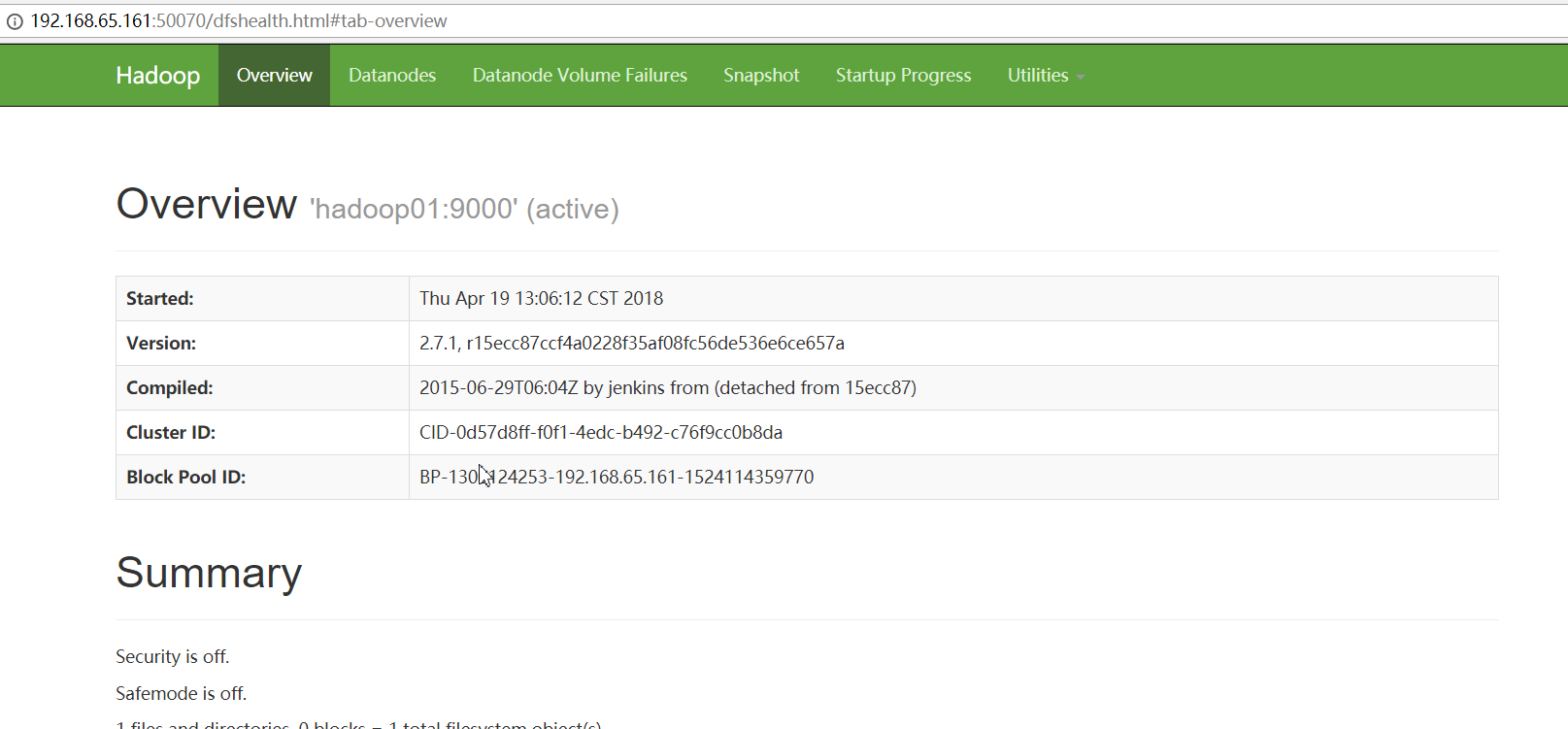
停止

stop-all.sh

1. 测试



访问hadoop01:50070

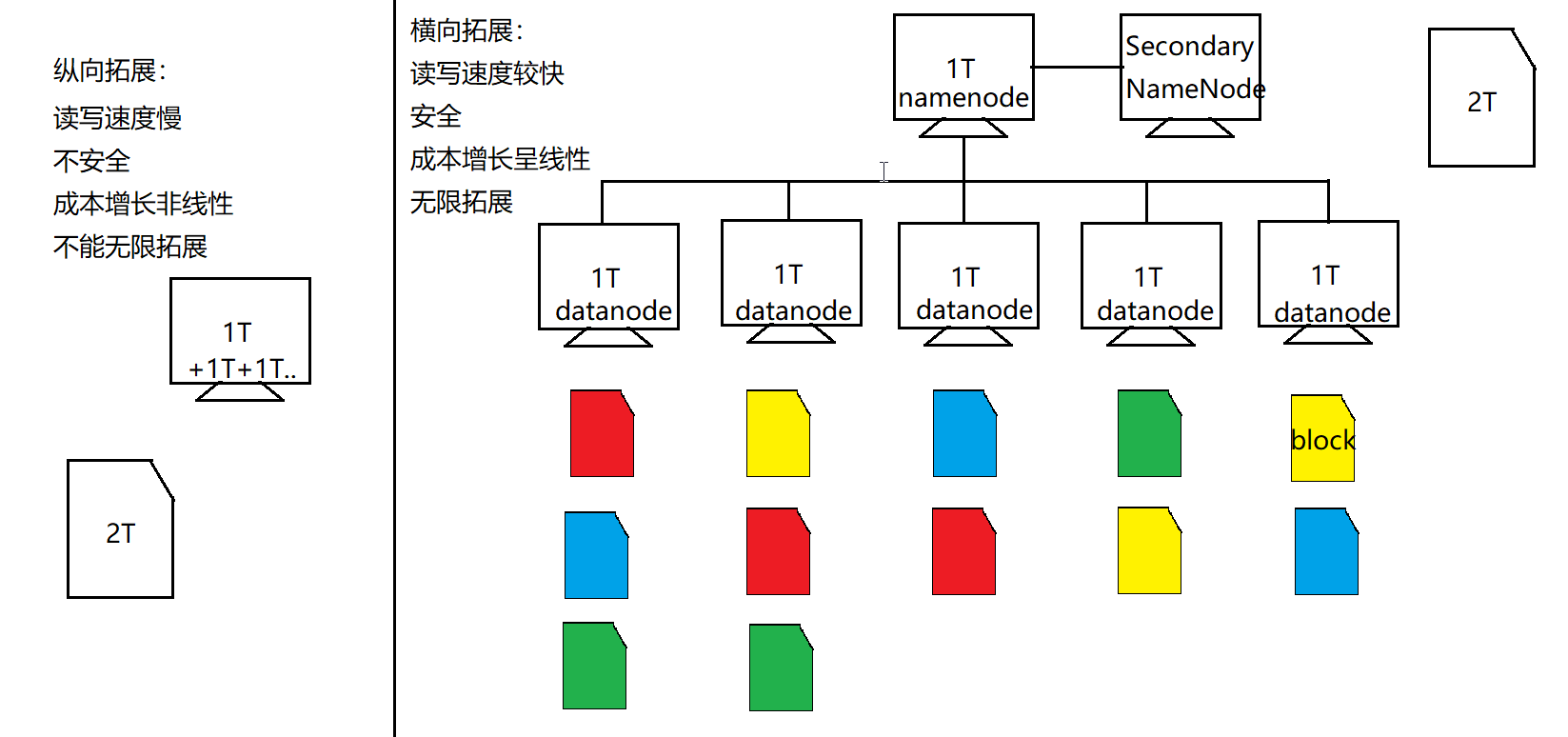


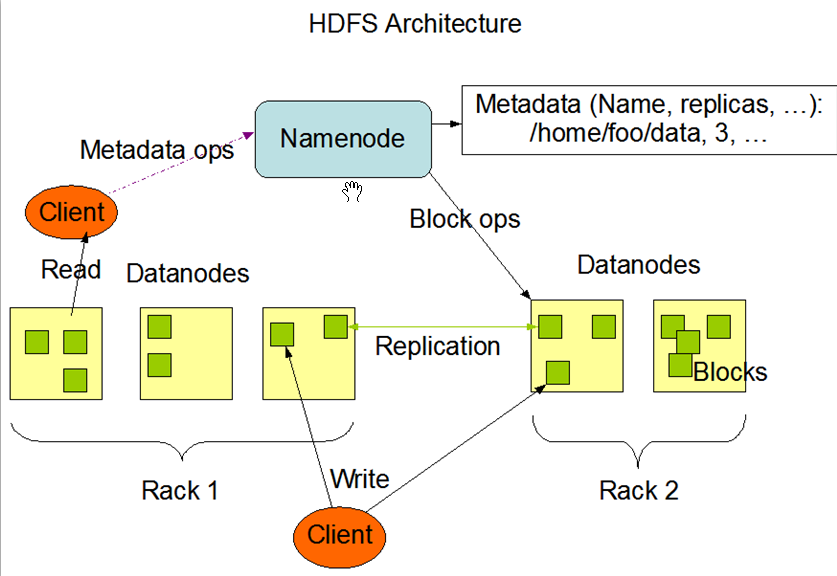
安装成功！

## HDFS详解

HDFS：分布式文件存储系统。用来存储海量数据。

思考：有一个超大文件让我们存储，该怎么做？





### NameNode

HDFS集群中的老大，负责元数据信息（文件分为几块，备份几份，每一份都存在哪里的描述信息）的存储和整个集群工作的调度。

### DataNode

集群中干活的小弟，存放文件块，记录自己存放文件的基本信息。

### Block

文件块，Hadoop1.0时，每块64M。Hadoop2.0时，每块128M。默认备份三份。

### SecondaryNameNode

NameNode的小秘，帮助NameNode干一些其他的事情。分担NameNode的压力。

### HDFS优点

可以存储超大文件（无限拓展）

高容错，支持数据丢失自动恢复

可以构建在廉价机上

### HDFS缺点

做不到低延迟访问

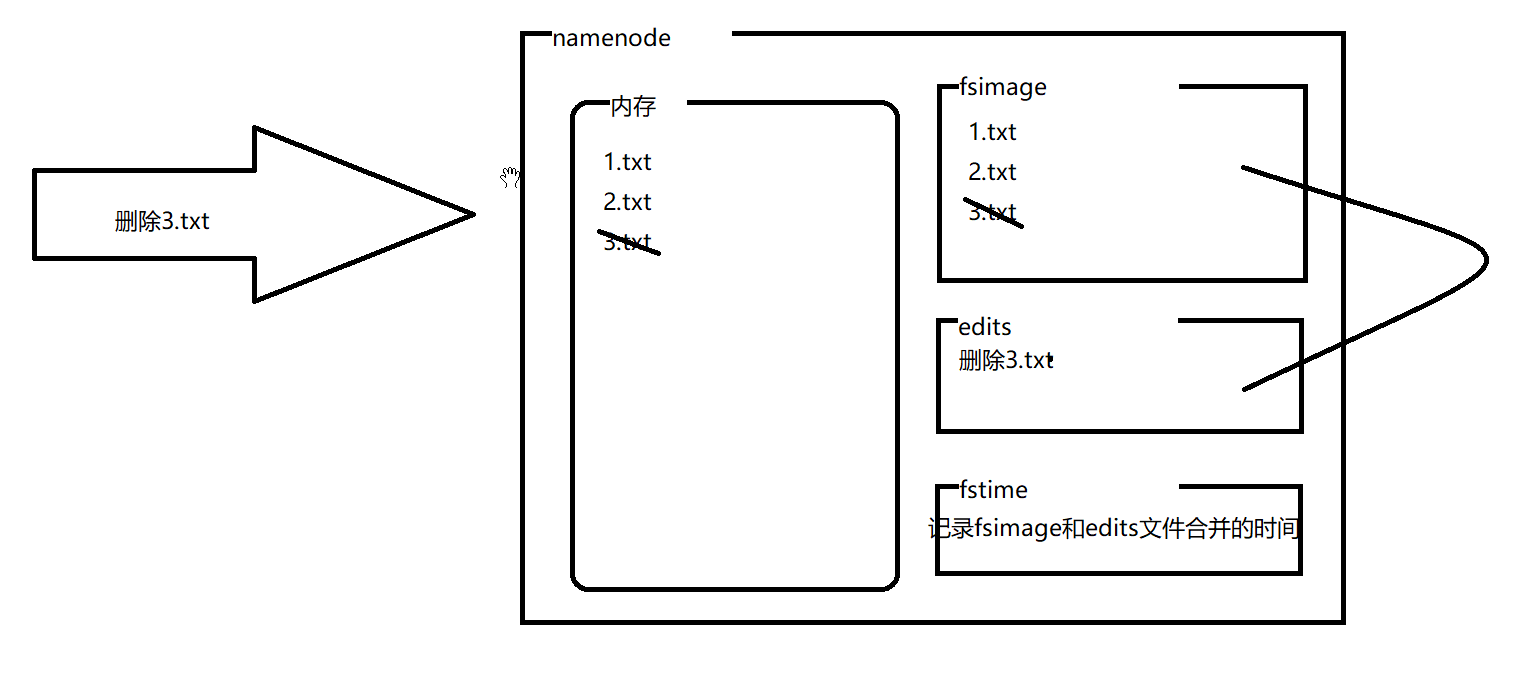
不支持超强的事务

不适合存大量小文件

不支持行级别的增删改

## HDFS细节

### NameNode、SecondaryNameNode如何工作？

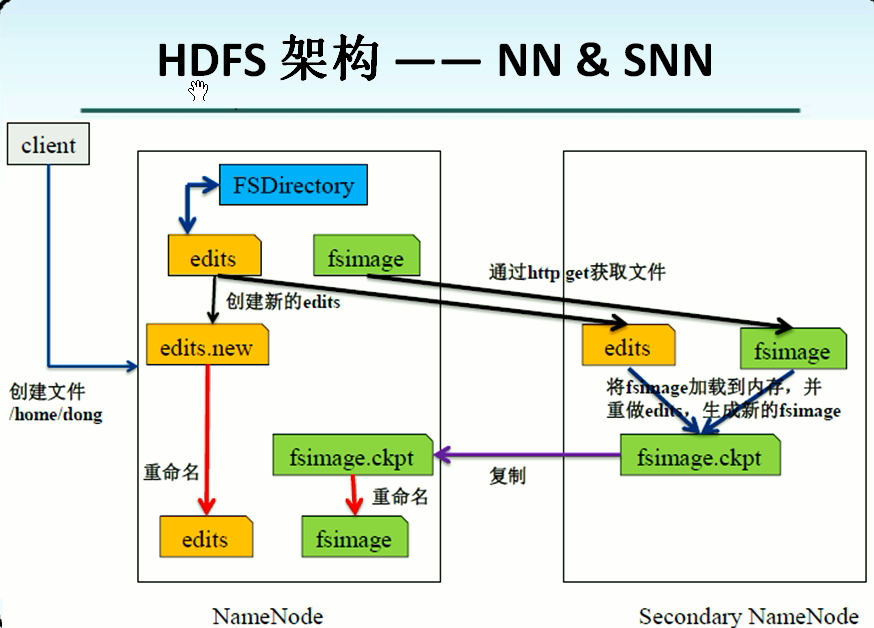


NameNode自己完成文件合并存在的问题：

Edits文件在合并时被占用，造成此时HDFS无法对外提供服务。

NameNode本来就是集群中最忙的节点，不时的合并文件加大了他的压力，NameNode一旦宕机，整个系统瘫痪，磁盘损坏会造成整个系统中的所有文件丢失。

所以引入了SecondaryNameNode来帮助NameNode完成文件合并的工作。



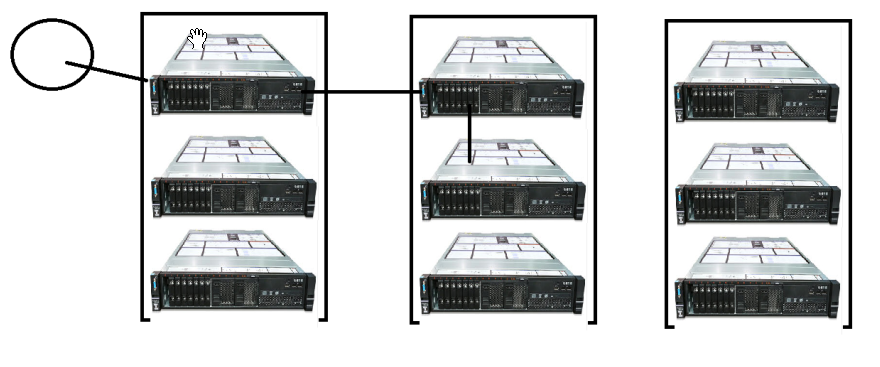
### Blok备份如何放置？

第一份：如果该文件本身从Hadoop集群中的某个节点上传，那么第一份存放在上传节点中，如果从Hadoop集群之外上传，那么存放在相对不太忙，负载较小的节点上。

第二份：放在与第一份所放置的节点相邻机架上的某个节点上。

第三份：放在与第二份所在节点的机架上的另外一台机器上。

三份以上，放置在负载相对较小的节点上。



## HDFS基本Shell操作

创建文件夹（不支持多级创建）：

hadoop fs -mkdir /xxx

查看目录：

hadoop fs -ls /xxx

递归查看多级目录：

hadoop fs -lsr /xxx

上传文件到HDFS：

hadoop fs -put xxx.txt /xxx

下载文件到本地当前目录：

hadoop fs -get /xxx/xxx/xxx.txt

删除文件：

hadoop fs -rm /xxx/xxx/xxx.txt

删除文件夹(文件夹必须为空)：

hadoop fs -rmdir /xxx/xxx

强制删除文件夹或文件

Hadoop fs -rm -r /xxx

## MapReduce

分布式计算系统

## Yarn

Hadoop中的大管家，负责整个集群的资源管理调度。主要用于管理MapReduce相关资源。

原来HDFS中的数据只能被MapReduce直接处理，引入Yarn之后可以支持多种数据处理工具的接入，包括Spark等（相当于插排）。

