## Hadoop基本知识介绍

**Hadoop是什么：分布式存储+分布式计算平台； 适合一次写入多次读取，顺序读写，不支持多用户并发写相同文件。Hadoop分布式文件系统(HDFS)分布在集群内多台机器上。使用适度的复制，集群可以并行读取数据，进而提供很高的吞吐量。这样一组通用机器比一台高端服务器更加便宜。代码向数据迁移的理念被应用在Hadoop集群自身。这种理念符合Hadoop面向数据密集型处理的设计目标。要运行的程序代码在规模上比数据小几个数量级，更容易移动。此外，在网络上移动数据要比在其上加载代码更花时间。不如让数据不动，而将可执行代码移动到数据所在的机器上去。通过采用分布式存储、迁移代码而非迁移数据， Hadoop在处理大数据集时避免了耗时的数据传输问题。Hadoop尽量在计算节点上存储数据，实现数据本地快速访问，数据本地化是Hadoop处理数据的核心。在Hadoop中，数据的来源可以有任何形式，但最终会转化为键值对以供处理。**

Hadoop由4个主要构成部分：

1) HDFS：分布式存储

\* 分布式文件存储系统；

\* 提供了高可靠性、高扩展性和高吞吐率的数据存储服务；

\* HDFS典型结构：物理结构+逻辑结构；

2) MapReduce：分布式计算引擎

\* 分布式计算框架（计算向数据移动-->移动代码而非移动数据）

\* 具有易于编程、高容错性和高扩展性的优点

3) YARN：资源调配模块（引擎）（分布式资源管理框架）

\* 负责集群资源的管理和调度

4）Hadoop Ozone： Hadoop的对象存储。（后来新增）

**关于data blocks默认大小的说明：**

从Hadoop的官网上看了各个版本的说明文档中关于 Data Blocks 的说明，发现是从2.7.3版本开始，官方关于Data Blocks 的说明中，block size由64 MB变成了128 MB的。

### 三种运行模式：

**本地（独立）模式**：

Hadoop配置为以非分布式模式运行，作为单个Java进程。这对调试很有用

**伪分布式模式：**

Hadoop还可以在伪分布式模式下在单节点上运行，其中每个Hadoop守护程序在单独的Java进程中运行，用不同的Java进程模仿分布式运行中的各类结点；在一台机器上运行HDFS文件系统，运行MapReduce程序，从HDFS上获取数据，结果存放到HDFS上

**完全分布式模式：**

运行在多台机器上，同时只有一个HDFS系统

### HDFS的设计理念

硬件故障是常态而非例外。HDFS实例可能包含数百或数千台服务器计算机，每台计算机都存储文件系统数据的一部分。事实上，存在大量组件并且每个组件具有一定的的故障概率，意味着HDFS的某些组件始终不起作用。因此，检测故障并从中快速自动恢复是HDFS的核心架构目标。在HDFS上运行的应用程序需要对其数据集进行流式访问。HDFS设计用于批处理而不是用户的交互式使用。

重点是数据访问的高吞吐量而不是数据访问的低延迟。

注释：批处理是指按照预定的方式运行数据查询。

注释：“流”就是源源不断地，分不清开始和结束的字节，每次的收发量不确定；“报文”就是每次的收发都有最大的长度限制(即报文长度)，接收端每次都会接收到一个报文，也就能分清每次报文的开始和结束。

### HDFS构成

**HDFS存储模型**

\* 文件线性切割成Block：偏移量（offset）

\* Block分散存储在集群节点中

\* 单一文件Block大小一致，文件与文件可以不一致

\* Block可以设置副本数，副本分散在不同的节点中？可以在不同节点吗？

\* 副本数不要超过节点数量

\* 文件上传可以设置Block大小和副本数

\* 已上传的文件Block副本数可以调整，大小不变

\* 只支持一次写入多次读取，同一时刻只有一个写入者

\* 只能追加，不能修改

**HDFS架构模型**

\* 文件的元数据（metadata）和文件数据是分开存储

\* （主）NameNode存储文件元数据

\* （从）DataNode存储文件数据

\* DataNode与NameNode保持心跳，由dataNode提交Block列表

\* HdfsClient（用户）与NameNode交互元数据信息

\* HdfsClient（用户）与DataNode交互文件数据信息

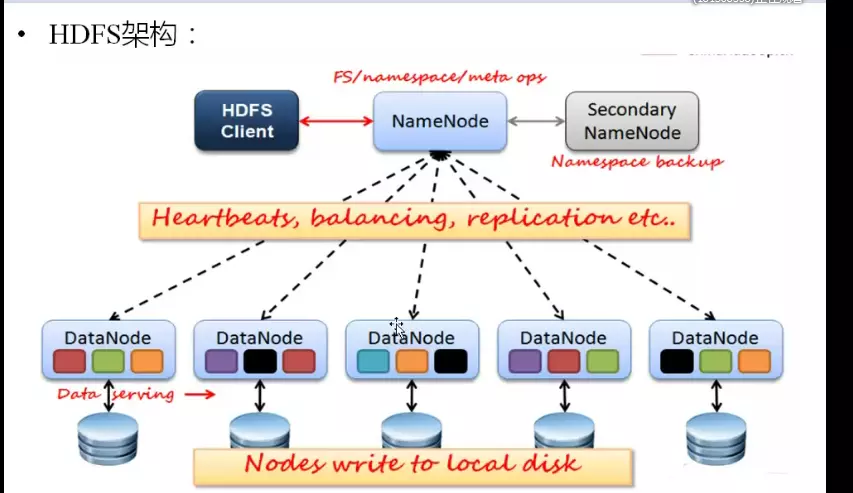
HDFS由3种主要的节点构成：

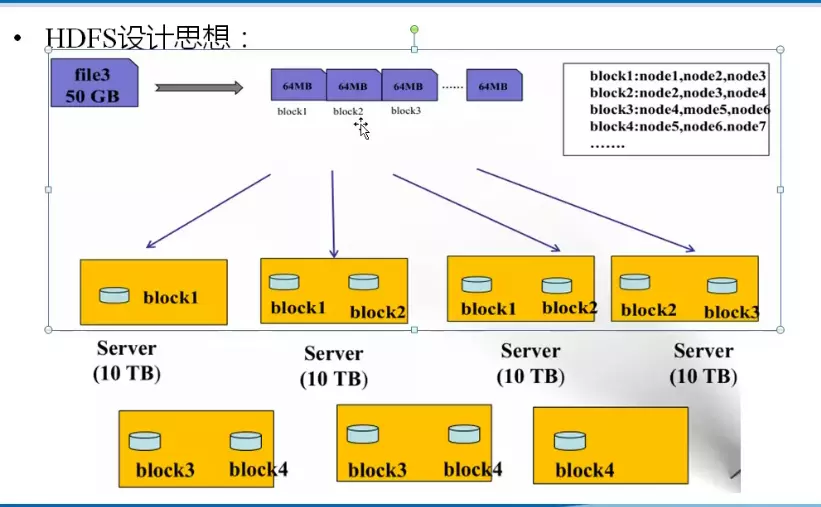
1) NameNode：用来存储数据的存放位置等元数据（不存放数据）；

2) DataNode：只用来存储数据；

3) SecondaryNameNode：辅助NameNode运行，将NameNode产生的元数据持久化到磁盘上；定期同步元数据映像文件和修改日志，NameNode发生故障时，备胎转正。

注意：生产环境中一台计算机通常只担任一种角色。测试环境中一台计算机可以担任多种角色；如伪分布模式中担任全部角色，测试集群中也常见NameNode和Secondary NameNode在同一台计算机上；





### NameNode

\* 只存在内存中（除了初始化和持久化的时候跟硬盘打交道，其余时候全部在内存中操作）

\* 持久化操作（假设内存只有1G，现在数据有1.2G，则需要做持久化）

\* NameNode的metadata信息在启动后加载到内存

\* Metadata信息存储到fsimage文件中

\* edits记录对metadata的操作日志（类似redis）

**注释：**

1) namenode存储数据的分布位置、数据的各种描述信息（如文件名、文件大小、文件所在目录、所有者名称、读写执行权限等）

2) 读数据时，要先从NameNode获取文件的分布位置（在哪些DataNode上），然后再从DataNode上读数据（当然，这个过程已经由HDFS的Shell或API实现了）

3) 写数据时，先向NameNode提交要写的文件的信息，NameNode检查自己的记录表，以找到合适的DataNode（需要找多个，因为每个文件还要创建副本）来存储这些数据，然后指挥这些DataNode串成一串接收数据。如果某个DataNode在接收数据时罢工，则忽略掉它（忽略之后造成的副本数少于指定数量会在后期补加副本）；如果所有选出的DataNode全挂掉（几率很低），则写入失败。

4) 需要时，可以调用NameNode列出HDFS中的文件夹及文件（如执行 hdfs dfs -ls / 命令时）。

**元数据存储位置**

具体存储位置由配置文件指定，即hdfs-site.xml中的 dfs.namenode.name.dir 指定这个位置可以用,列表指定多个位置进行扩容；

dfs.namenode.name.dir的默认值是<file://${hadoop.tmp.dir}/dfs/name>；

${hadoop.tmp.dir}的默认值是 /tmp/hadoop-${user.name}；

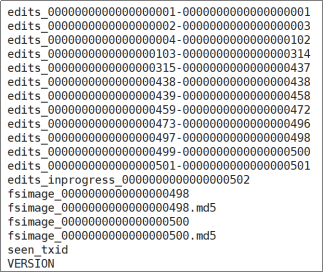
${user.name}是安装Hadoop的用户名；这些默认值在Hadoop官方网站的文档中有详细说明。配置时，为了简单，通常会在core-site.xml中修改 ${hadoop.tmp.dir}，因为这个值在各个具体配置文件中被多次引用！

通过Hadoop的浏览站点也能找到NameNode元数据的存储位置，在Overview页上



**持久化的元数据**

进入NameNode存放元数据的目录，可以看到如下文件：



NameNode的元数据运行时是加载在内存中的（因为这些数据需要快速查询），当NameNode关闭时，内存中的元数据又会持久化到硬盘上（上图中的fsimage文件）。在不考虑HDFS的写操作时，元数据只在内存和硬盘之间交换。

内存元数据 == fsimage

但HDFS集群是要写数据的，这时就会导致内存中的元数据和硬盘上的fsimage文件不同。内存是易失的，如意外断电或死机都会导致内存中的数据丢失，为了保证所有的数据变化都能记录下来，HDFS会将所有的操作都记录到edits文件中。这样在写数据时就会形成以下关系：

内存元数据 == fsimage + edits

### SecondaryNameNode功能

生产环境中NameNode会长时间运行，几乎不关闭。而内存中的元数据是在NameNode关闭时进行持久化的(即持久化在磁盘上面)。这样生产环境中内存元数据与持久化(即磁盘上的)元数据差异就会越来越大，会导致生成大量的edits文件，NameNode重启速度变慢。为了解决这些问题就产生了Secondary NameNode。

SecondaryNameNode会将NameNode中产生的fsimage和edits下载过来，执行合并生成新的fsimage，然后把新的fsimage再重新发回NameNode，这样NameNode中的持久化元数据就能与内存中的元数据一致或非常接近了，edits文件的数量大大减少。

**DataNode**

通过向NameNode发送心跳信息保持与其联系（每3秒一次），如果NameNode 10分钟没有收到DataNode的心跳，则认为其已经lost，则将其block信息copy到其他DataNode上。

**Block副本的放置策略**

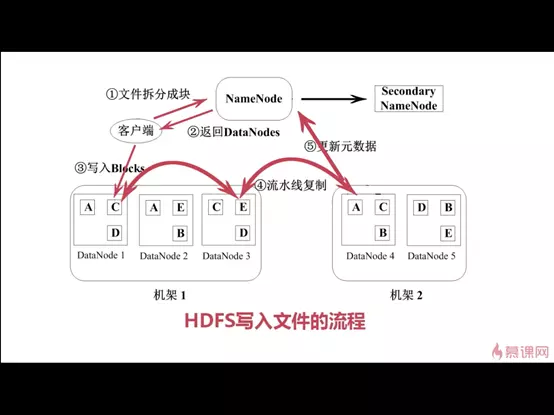
第一个副本：放置在上传文件的datanode上，如果是集群外提交，则随机挑选一台磁盘不太满，cpu不太忙的节点；

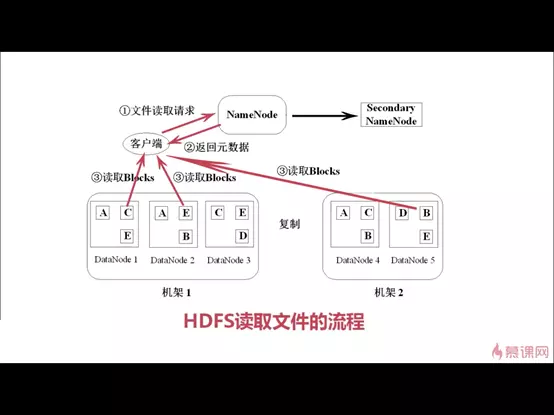
第二个副本：放置在第一个副本不同的机架的节点上；

第三个副本：与第二个副本相同的机架的节点；

更多副本：随机节点；

### HDTS写入文件的流程**：**



HDTS读取文件的流程：

### 基本命令：

hadoop fs –ls /

把当前目录下HDFS文件夹打印出来

hadoop fs –put tt.txt input/

把本地的一个文件tt.txt放到input目录下

hadoop fs –mkdir input

建立input目录

hadoop fs –get input/tt.txt tt1.txt

从hdfs下载文件到本地

hadoop dfsadmin –report

查看文件系统内所有信息