Hadoop 概论和快速入门

Hadoop 培训大纲

- Hadoop 源起和概论
  - 思考: Hadoop的核心思想
- Hadoop 简介：缘起与术语
  - 练习一：Hadoop 单机安装
- HDFS 理念与命令解说
  - 练习二：HDFS 操作实务
- MapReduce 简介:
  - 练习三：MapReduce 范例操作

Hadoop 培训大纲

Hadoop 集群安装配置:

- 练习四：Hadoop 集群安装实战

- Hadoop 相关应用(1) -Hadoop Streaming
  - 练习五：Hadoop Streaming 操作练习

Hadoop 安装和配置

名词解释

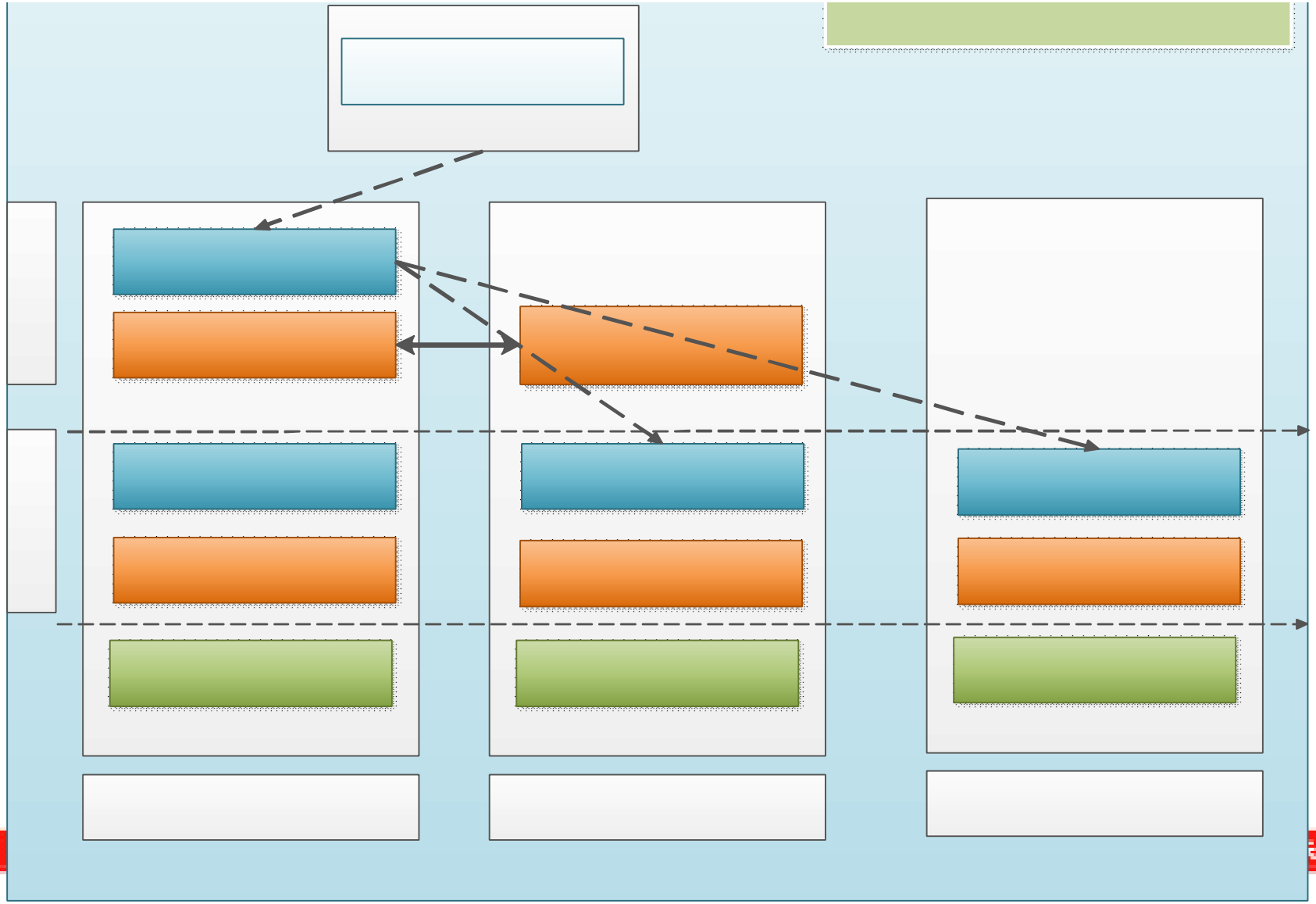
- Hadoop, Apache开源的分布式框架。源自Google GFS,BigTable,MapReduce 论文。

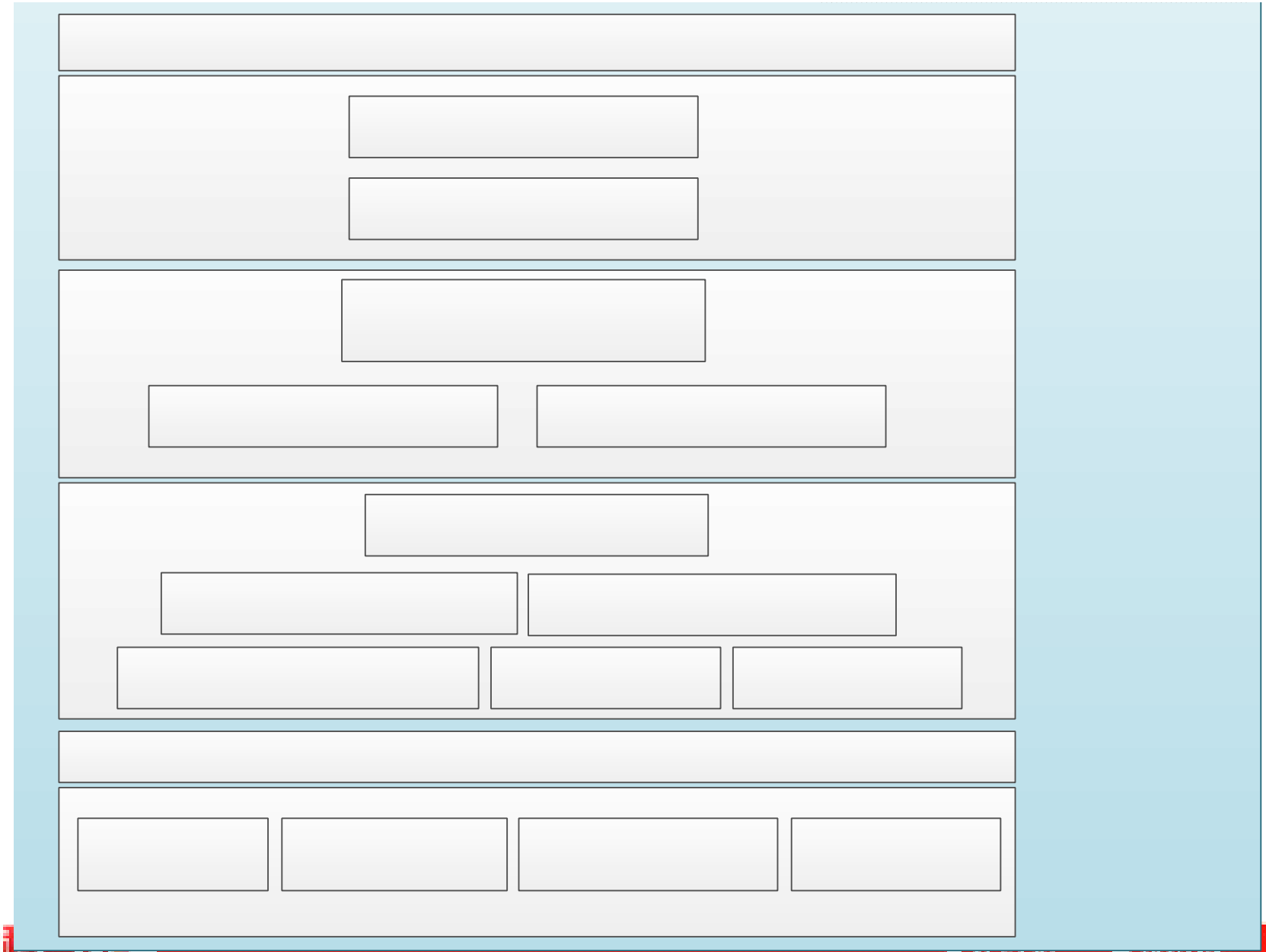
== HDFS ==

- HDFS (Hadoop Distributed File System),Hadoop 分布式文件系统。
- NameNode,HDFS命名服务器,负责与DataNode文件元信息保存。
- DataNode,HDFS数据节点，负责存储数据存储并汇报给NameNode。
- **SecondaryNamenode,NameNode的镜像备份节点**

- **==Map Reduce==**
- JobTracker，hadoop的Map/Reduce调度器，负责与TackTracker通信分配计算任务并跟踪任务进度。
- TaskTracker,启动和管理Map和Reduce子任务的节点。

Hadoop简易版本 总体组件架构





Hadoop 单机部署（Hadoop Pseudo-Distributed Mode）

----- 配置系统环境------

- Step 0: 配置ip和用户,hostname,本地DNS 服务 /etc/hosts
- Step 1: 配置ssh无密钥登陆 (Setup SSH key exchange)
- Step 2. 安装JDK,配置环境变量 (Install JDK)

------ 配置 hadoop 环境-----

- Step 3: 下载软件 (Download Hadoop Source Package)
- Step 4: 配置hadoop 环境变量 (Configure hadoop-env.s)
  - export JAVA_HOME=/usr/java/jdk1.6.0_21/

- Step 5:配置Hadoop配置(Configure core-site.xml hdfs-site.xml)
  - Set Namenode to hdfs://master:9000
  - Set Jobtracker to master:9001

Hadoop 单机部署（Hadoop Pseudo-Distributed Mode）

在主节点上修改hadoop配置文件 详细说明

Hadoop的配置文件存放在hadoop安装目录下的conf目录中，主要有以下几个配置文件要修改：

- conf/hadoop-env.sh：Hadoop环境变量设置
- conf/core-site.xml：主要完成NameNode的IP和端口设置
- conf/hdfs-site.xml：主要完成HDFS的数据块副本等参数设置
- conf/mapred-site.xml：主要完成JobTracker IP和端口设置
- conf/masters：完成master节点IP设置
- conf/slaves：完成Slaves节点IP设置

注：这个过程仅需在主节点上进行，然后将随着主机上安装好的Hadoop目录一起复制到所有从节点

Hadoop 单机部署（Hadoop Pseudo-Distributed Mode）

• Step 6: Format HDFS

格式化文件系统 bin/hadoop namenode -format

• Step 7: Start Hadoop

------ 启动Hadoop集群----

– bin/start-all.sh

• Step 8: 检查Hadoop状态

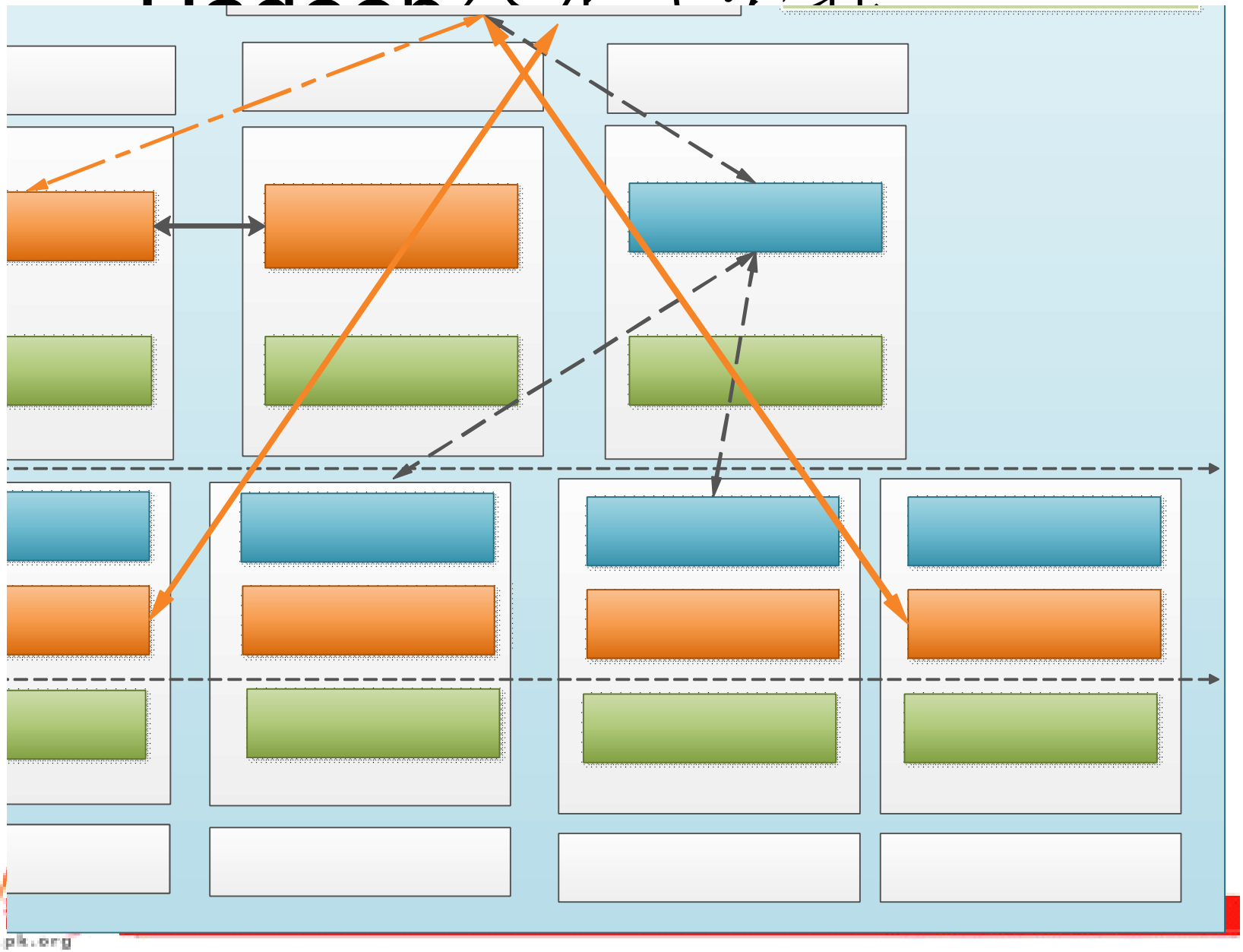
– 检查任务 http://master:50070/

– 检查任务 http://master:50030/

测试:

hadoop jar hadoop-*-examples.jar pi 10 100

Hadoop分布式安装



Hadoop 集群部署-练习

- 启动命名节点 /opt/hadoop/bin/hadoop-daemon.sh start namenode
- 启动第二命名节点 /opt/hadoop/bin/hadoop-daemon.sh start secondarynamenode
- 启动数据节点 /opt/hadoop/bin/hadoop-daemon.sh start datanode
- 启动 任务子节点 /opt/hadoop/bin/hadoop-daemon.sh start jobtracker
- 启动 任务子节点 /opt/hadoop/bin/hadoop-daemon.sh start tasktracker

- 检查挂载健康情况
- jps
- http://localhost:50030/
- http://localhost:50070/

- 注意启动和关闭集群顺序

使用Hadoop计算pi

- 蒙地卡罗算法
  (http://zh.wikipedia.org/wiki/%E8%92%99%E5%9C%B0%E5%8D%A1%E7%BE%85%E6%96%B9%E6%B3%95)

- 蒙特卡罗方法（Monte Carlo method），也称统计模拟方法，是二十世纪四十年代中期由于科学技术的发展和电子计算机的发明，而被提出的一种以概率统计理论为指导的一类非常重要的数值计算方法。是指使用随机数（或更常见的伪随机数）来解决很多计算问题的方法。

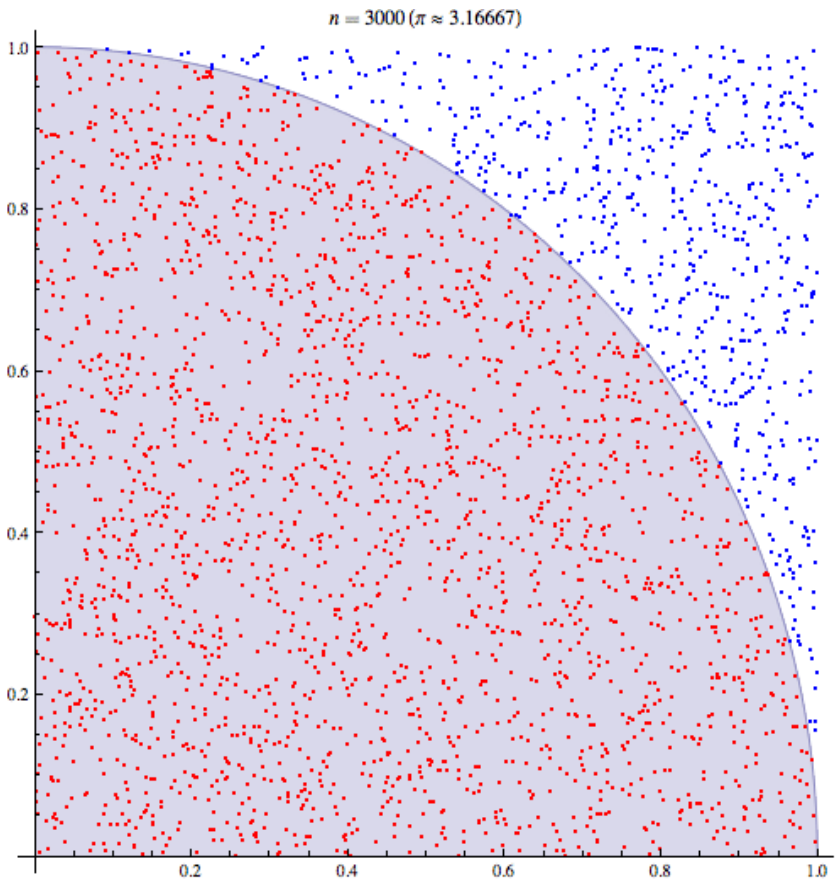
- 20世纪40年代，在John von Neumann，Stanislaw Ulam和Nicholas Metropolis在洛斯阿拉莫斯国家实验室为核武器计划工作时，发明了蒙特卡罗方法。因为Ulam的叔叔经常在蒙特卡罗赌场输钱得名，而蒙特卡罗方法正是以概率为基础的方法。与它对应的是确定性算法。

- 蒙特卡罗方法在金融工程学，宏观经济学，生物医学，计算物理学（如粒子输运计算、量子热力学计算、空气动力学计算）等领域应用广泛。

运行一个MapReduce程序

运行PI计算程序

```
hadoop jar hadoop-*-
    examples.jar pi 10
    100
```



Hadoop 快速入门

(mapreduce+hdfs)

思考 100TB 级别数据如何存储 ？

HDFS: 开发动机

高效:

- 提供 高稳定,高效的存储方案。

廉价:

- 使用廉价服务器,构建整体的海量存储方案。

可扩展

- 整体存储能力和硬件新增弹性扩展

HDFS 设计原则

- 文件以块(block)方式存储
- 每个块带下远比多数文件系统来的大(预设64M) linux 4k大小
- 通过副本机制提高可靠度和读取吞吐量
- 每个区块至少分到三台DataNode上
- 单一 master (NameNode)来协调存储元数据(metadata)
- 客户端对文件没有缓存机制 (No data caching)

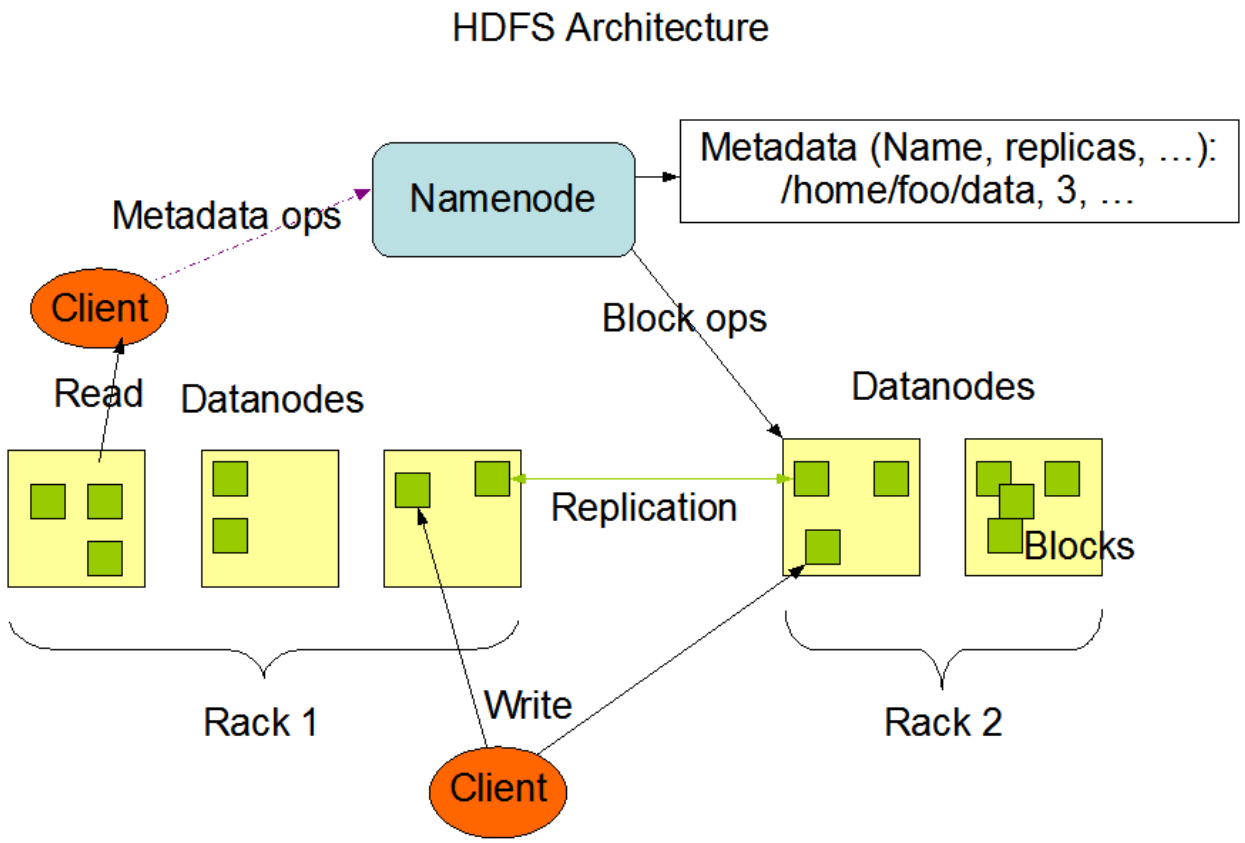
HDFS能做什么？

- 存储并管理PB级数据
- 处理非结构化数据
- 注重数据处理的吞吐量（latency不敏感）
- 应用模式为：write-once-read-many存取模式

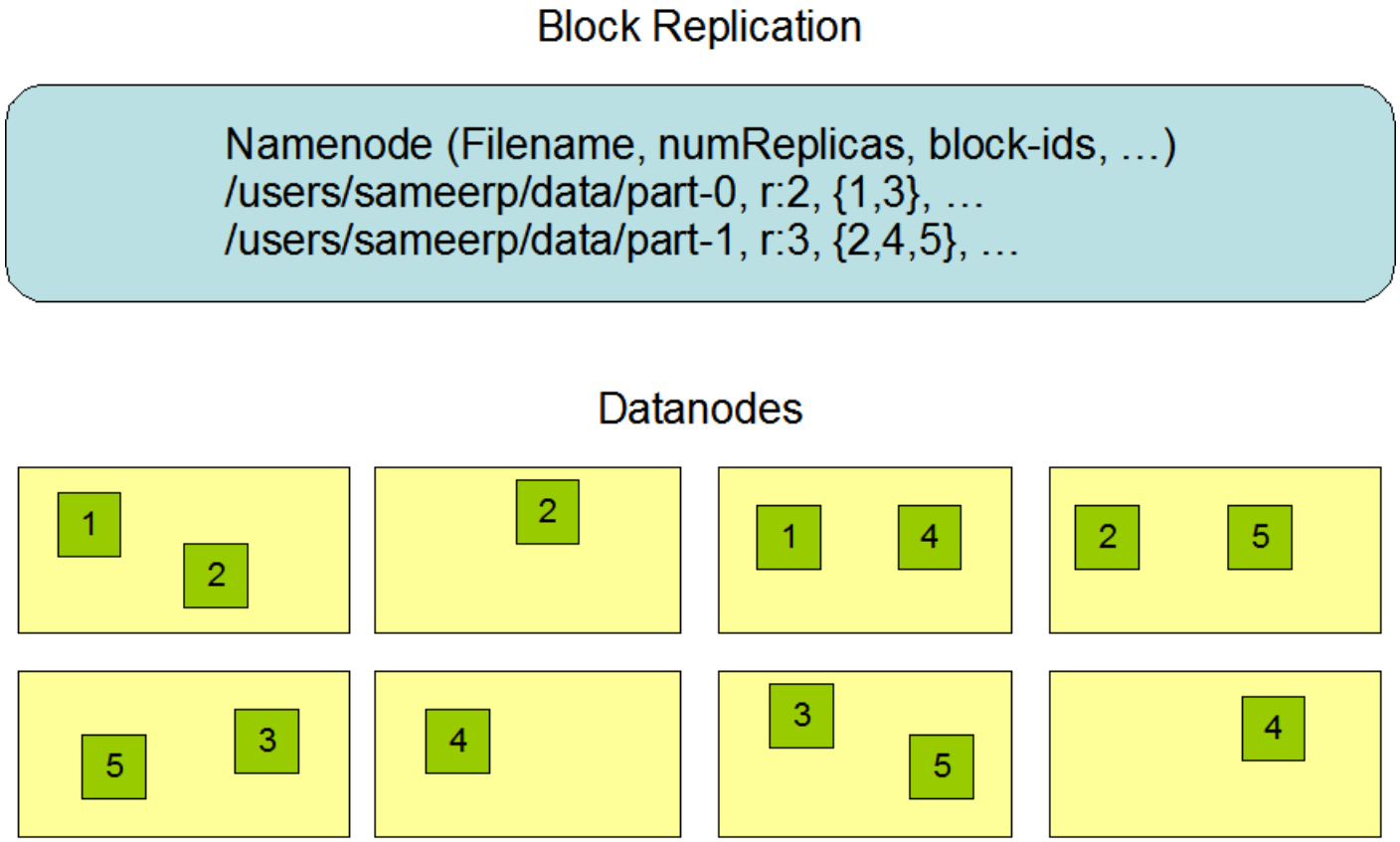
HDFS不适合做什么？

- 存储小文件 (不建议使用)
- 大量的随机读 (不建议使用)
- 需要对文件的修改 (不支持)

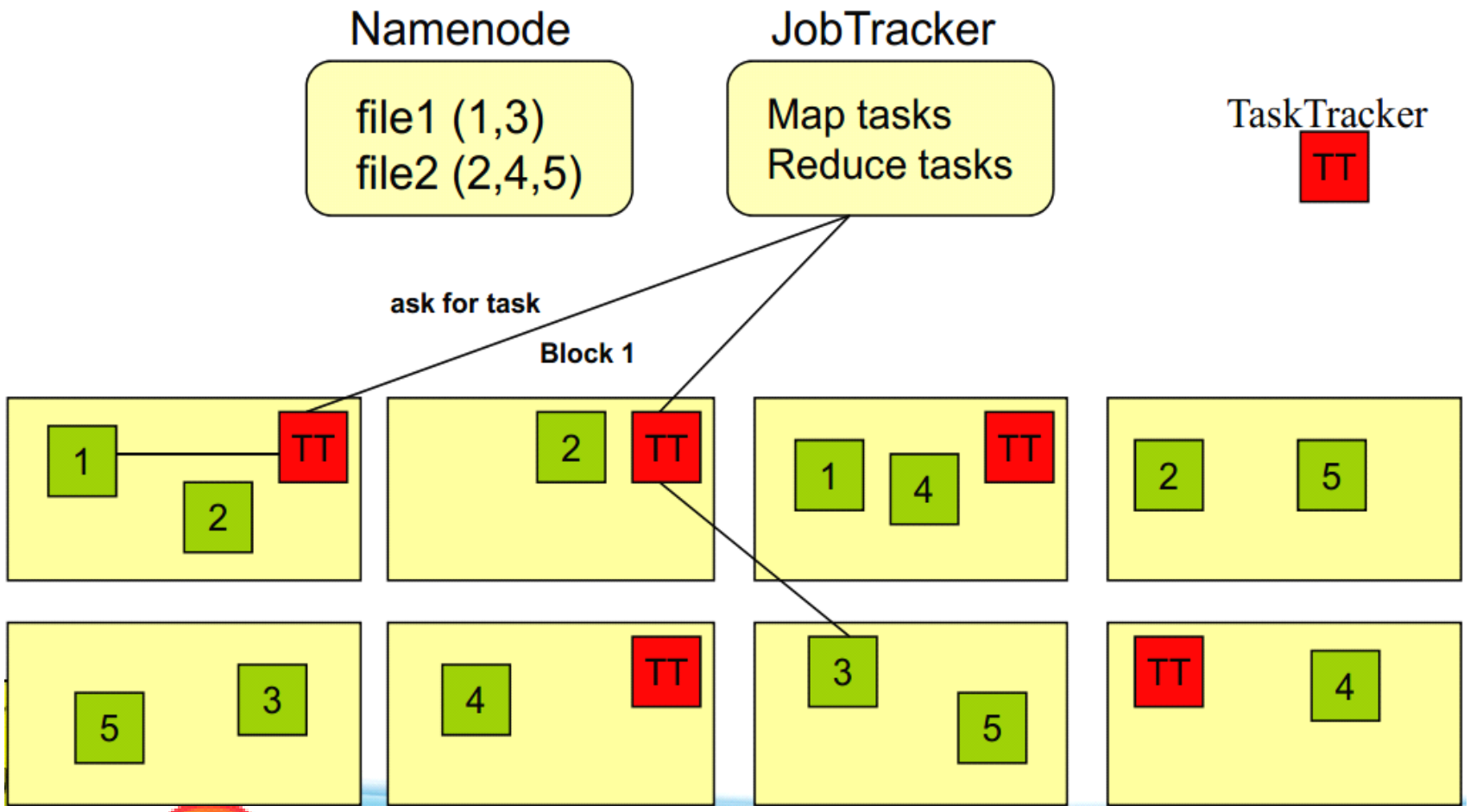
HDFS 基本结构


HDFS Architecture
Namenode
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Metadata ops
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

HDFS 分片复制


Block Replication
Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …
Datanodes
1
2
2
1
4
2
5
5
3
4
3
5
4

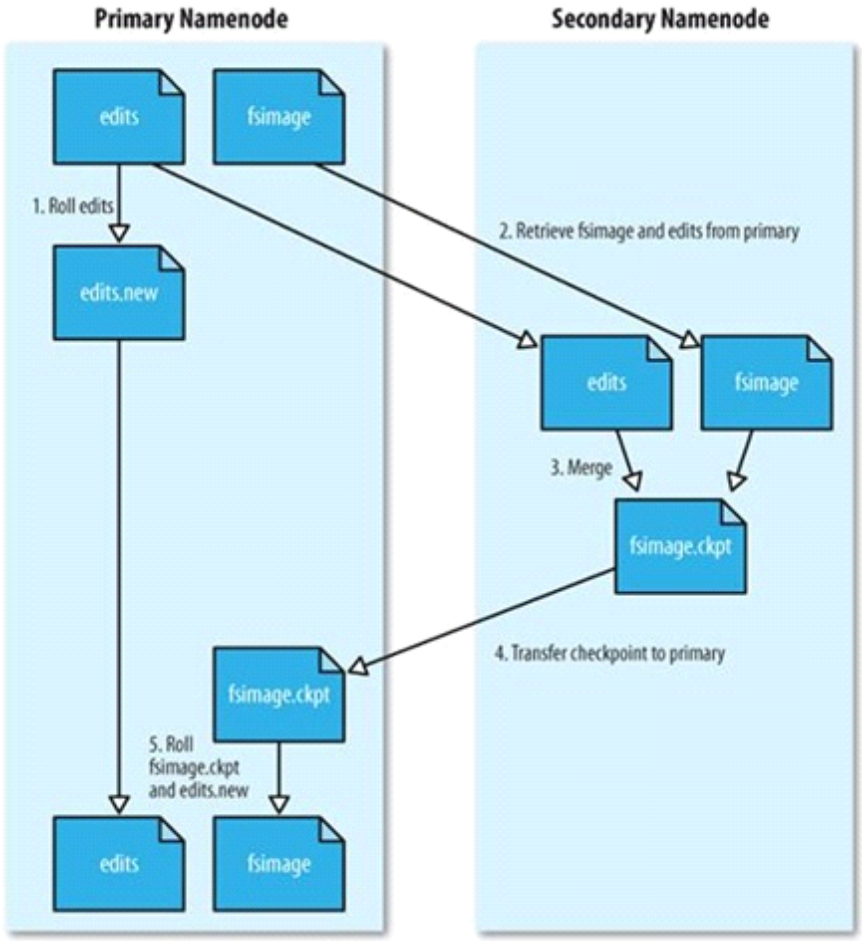
Task和HDFS交互



NameNode(NN)

- NameNode主要功能提供名称查询服务，它是一个jetty服务器
- NameNode保存metadate信息包括
  - 文件owership和permissions
  - 文件包含哪些块
  - Block保存在哪个DataNode（由DataNode启动时上报）
- NameNode的metadate信息在启动后会加载到内存
- metadata存储到磁盘文件名为”fsimage”
  - Block的位置信息不会保存到fsimage

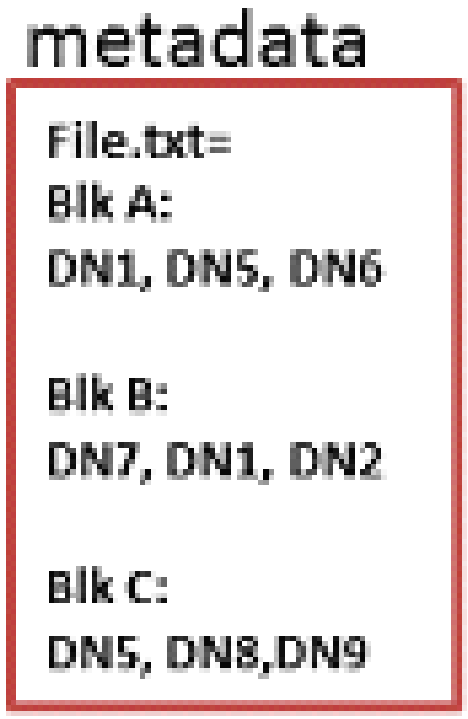
NameNode(NN)和Secondary NameNode(SNN)



Secondary NameNode定期地合并fsimage和Edits日志文件，并保持Edits日志文件的大小在一个上限值内，由于它的内存需求与NameNode的一致，所以它通常运行在NameNode以外的一台机器上。

Checkpoint配置参数:
fs.checkpoint.period，默认:1小时，指定连续2次创建Checkpoint的最大时间间隔。
fs.checkpoint.size，默认:64MB，当编辑日志大小到达该设置值，即使创建Checkpoint的最大时间间隔未到也强制促其执行创建Checkpoint。

NameNode



块存储结构

```
/data/cache1/dfs/nn/
|-- current
|   |-- VERSION
|   |-- edits
|   |-- fsimage
|   `-- fstime
|-- image
|   `-- fsimage
|-- in_use.lock
`-- previous.checkpoint
    |-- VERSION
    |-- edits
    |-- fsimage
    `-- fstime
```

metadate物理

DataNode（DN）

- 保存Block
- 启动DN线程的时候会向NN汇报block信息
- 通过向NN发送心跳保持与其联系（3秒一次），如果NN 10分钟没有收到DN的心跳，则认为其已经lost，并copy其上的block到其它DN

```
-bash-3.2$ sudo ls -l /data/cache1/dfs/dn/current/
total 594992
-rw-r--r--  1 hdfs hadoop     20495 Apr 29 21:11 blk_-1265257027172478037
-rw-r--r--  1 hdfs hadoop       171 Apr 29 21:11 blk_-1265257027172478037_1896.meta
-rw-r--r--  1 hdfs hadoop       282 Apr 29 21:10 blk_1355466200351616098
-rw-r--r--  1 hdfs hadoop        11 Apr 29 21:10 blk_1355466200351616098_1411.meta
-rw-r--r--  1 hdfs hadoop  32201859 Apr 29 21:13 blk_1363349205939896111
-rw-r--r--  1 hdfs hadoop    251587 Apr 29 21:13 blk_1363349205939896111_2355.meta
-rw-r--r--  1 hdfs hadoop     20349 Apr 29 21:11 blk_1486089392719042041
-rw-r--r--  1 hdfs hadoop       167 Apr 29 21:11 blk_1486089392719042041_1593.meta
-rw-r--r--  1 hdfs hadoop   1222666 Apr 29 21:12 blk_1841498388503862976
-rw-r--r--  1 hdfs hadoop      9563 Apr 29 21:12 blk_1841498388503862976_2595.meta
-rw-r--r--  1 hdfs hadoop   9250764 May 25 08:22 blk_1922063674315104428
-rw-r--r--  1 hdfs hadoop     72279 May 25 08:22 blk_1922063674315104428_32513112.meta
-rw-r--r--  1 hdfs hadoop   4509245 Apr 29 21:15 blk_-2039669051003294713
-rw-r--r--  1 hdfs hadoop     35239 Apr 29 21:15 blk_-2039669051003294713_4346.meta
```

Block的副本放置策略

- 第一个副本：放置在上传文件的DN；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：放置在于第一个副本不同的机架的节点上
- 第三个副本：与第二个副本相同集群的节点
- 更多副本：随机节点

再说Block

- 设置一个Block 64MB，如果上传文件小于该值，仍然会占用一个Block的命名空间（NameNode metadata），但是物理存储上不会占用64MB的空间
- Block大小和副本数由Client端上传文件到HDFS时设置，其中副本数可以变更，Block是不可以再上传后变更的

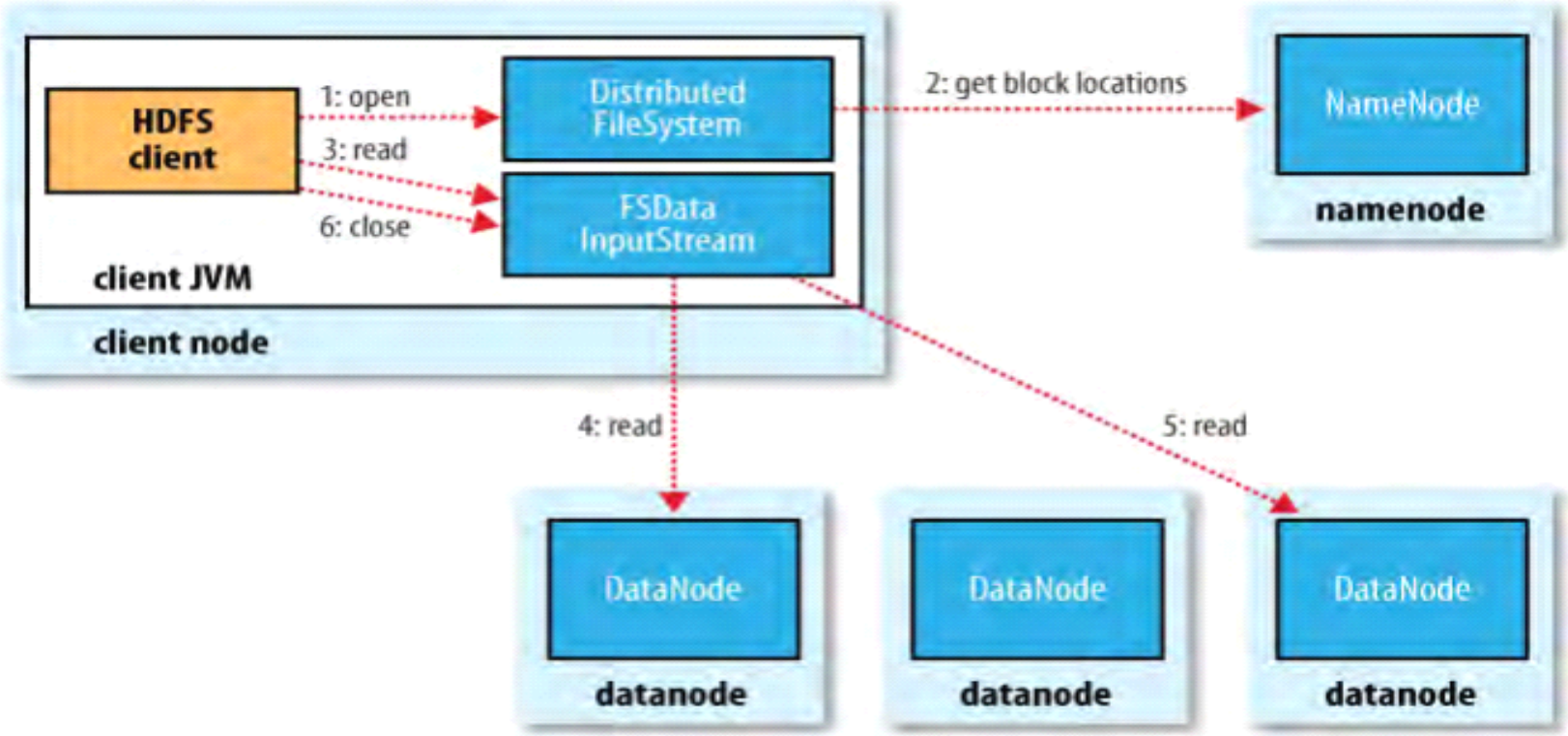
数据损坏(corruption)处理

- 当DN读取block的时候，它会计算checksum
- 如果计算后的checksum，与block创建时值不一样，说明该block已经损坏。
- client读取其它DN上的block；NN标记该块已经损坏，然后复制block达到预期设置的文件备份数
- DN在其文件创建后三周验证其checksum

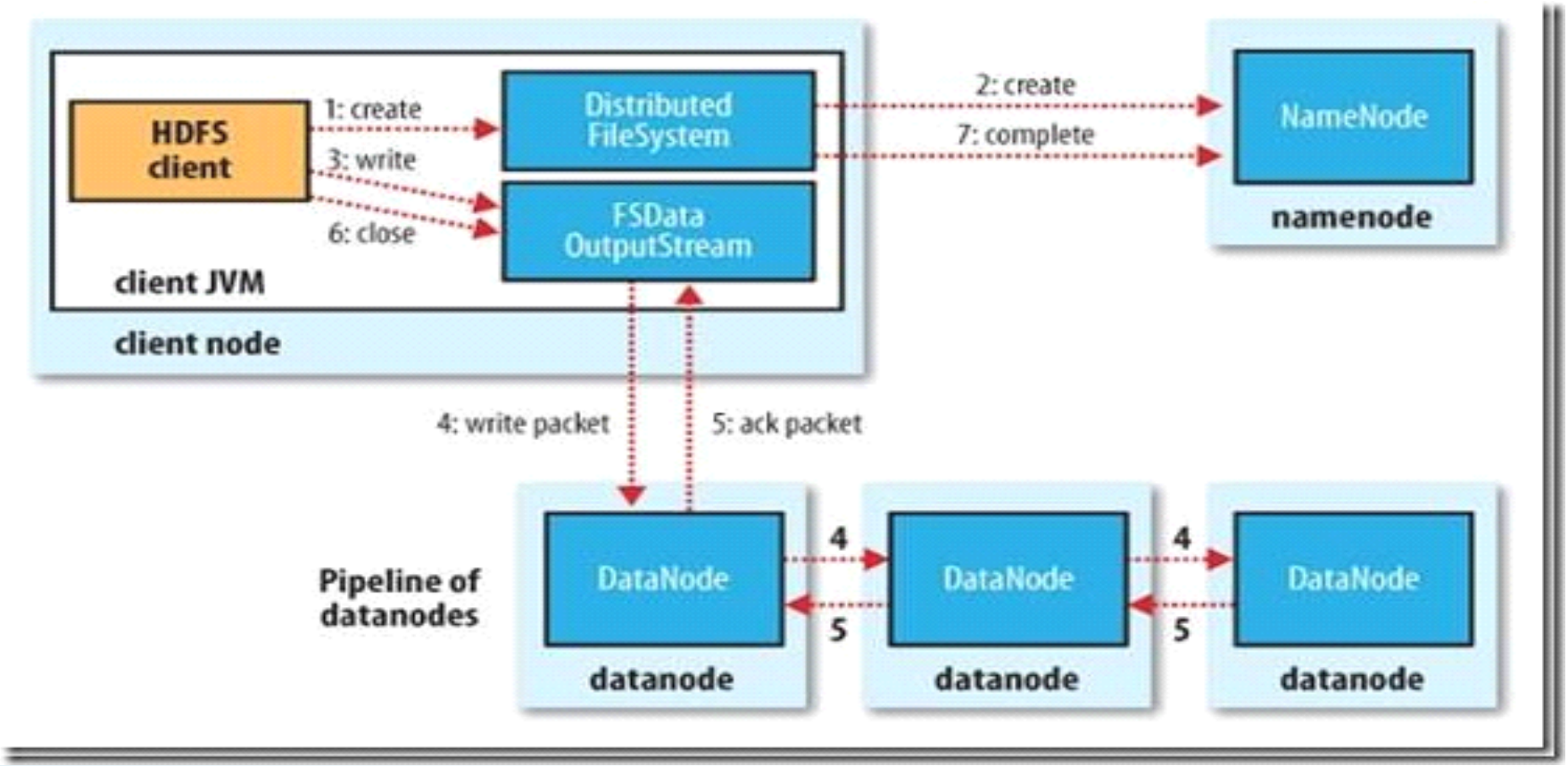
HDFS文件权限

- 与Linux文件权限类似
- r: read; w:write; x:execute，权限x对于文件忽略，对于文件夹表示是否允许访问其内容
- 如果Linux系统用户zhangsan使用hadoop命令创建一个文件，那么这个文件在HDFS中owner就是zhangsan
- HDFS的权限目的：阻止好人错错事，而不是阻止坏人做坏事。HDFS相信，你告诉我你是谁，我就认为你是谁

HDFS文件读取


HDFS client
1: open
3: read
6: close
Distributed FileSystem
FSData InputStream
client JVM
client node
2: get block locations
NameNode
namenode
4: read
5: read
DataNode
datanode
DataNode
datanode
DataNode
datanode

HDFS文件写入


HDFS client
1: create
3: write
6: close
Distributed FileSystem
FSData OutputStream
client JVM
client node
2: create
7: complete
NameNode
namenode
4: write packet
5: ack packet
Pipeline of datanodes
DataNode
4
5
datanode
DataNode
4
5
datanode
DataNode
datanode

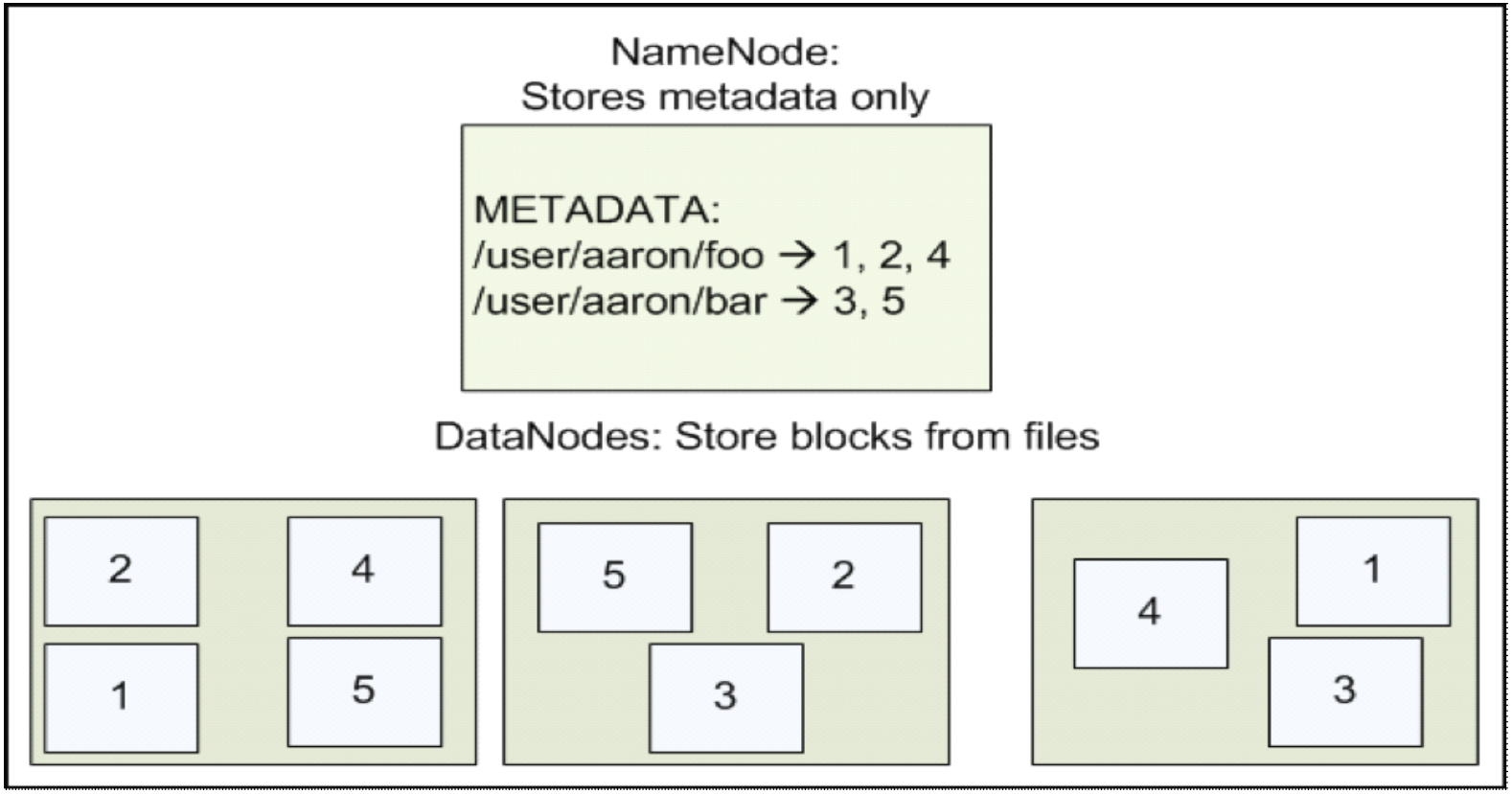
HDFS文件存储

两个文件，一个文件156M，一个文件128在HDFS里面怎么存储？

--Block为64MB

--rapliction默认拷贝3份

HDFS文件存储结构


NameNode:
Stores metadata only
METADATA:
/user/aaron/foo → 1, 2, 4
/user/aaron/bar → 3, 5
DataNodes: Store blocks from files
2
4
1
5
5
2
3
4
1
3

HDFS开发常用命令

- 创建一个文件夹？
- 上传一个文件？
- 删除一个文件和文件夹？
- 查看一个文件夹里面有哪些文件？
- 查看某个文件的内容？

Hadoop管理员常用命令

- hadoop job –list #列出正在运行的Job
- hadoop job –kill <job_id> #kill job
- hadoop fsck / #检查HDFS块状态，是否损坏
- hadoop fsck / -delete #检查HDFS块状态，删除损坏块
- hadoop dfsadmin –report #检查HDFS状态，包括DN信息
- hadoop dfsadmin –safemode enter | leave
- hadoop distcp hdfs://a:8020/xxx hdfs://b:8020/// #并行copy
- ./bin/start-balancer.sh #平衡集群文件