

Linux 大数据

NSD HADOOP

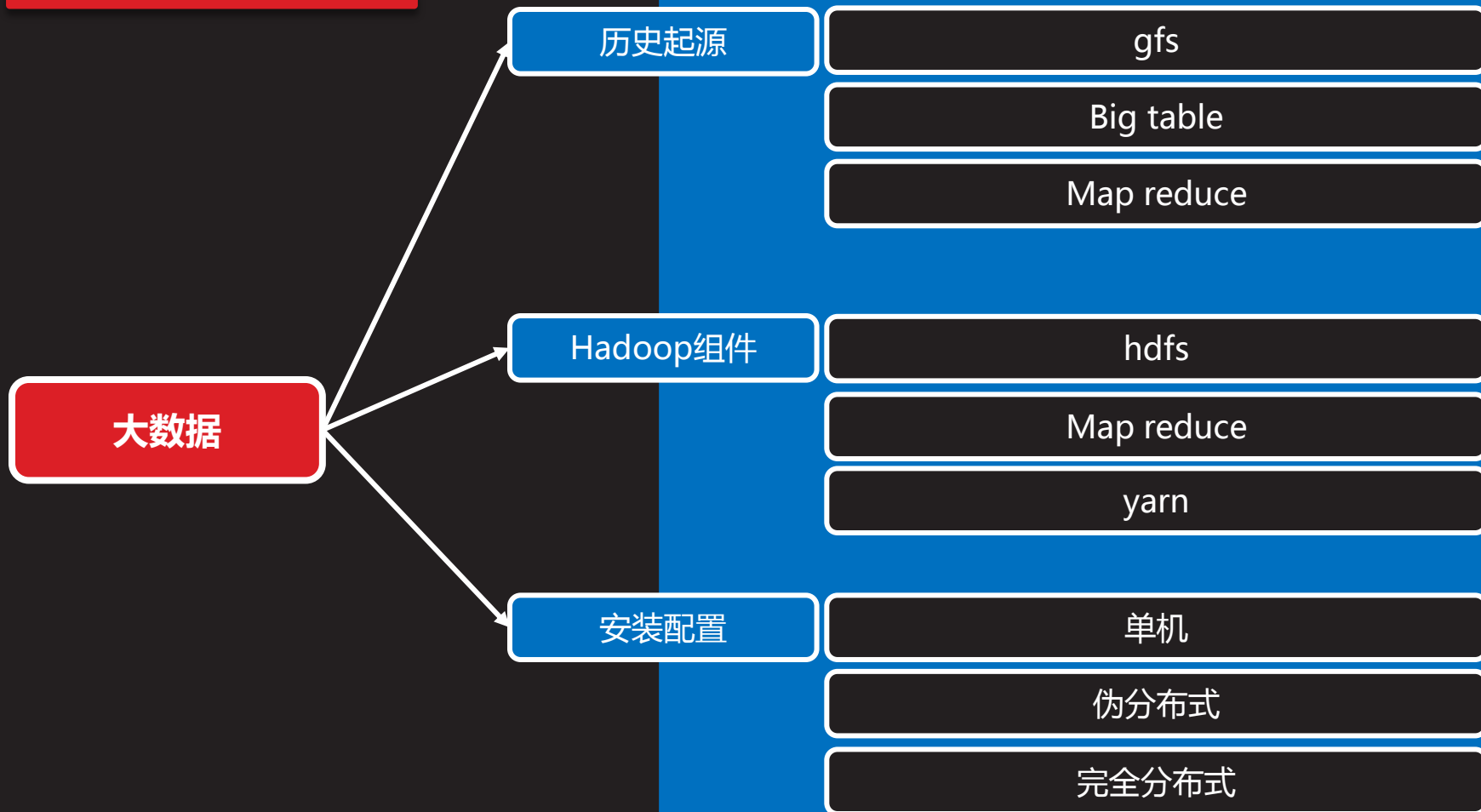
DAY01

内容

上午	09:00 ~ 09:30	什么是大数据
	09:30 ~ 10:20	大数据能做什么
	10:30 ~ 11:20	Hadoop历史起源
	11:30 ~ 12:00	Hadoop 组件介绍
下午	14:00 ~ 14:50	Hadoop安装配置
	15:00 ~ 15:50	
	16:10 ~ 17:00	Hadoop搭建练习
	17:10 ~ 18:00	总结和答疑



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，信息更是爆炸是增长，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在可接受时间下的收集、应用、管理和处理能力。
 - 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，可用于察觉商业趋势、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；这样的用途正是大型数据集盛行的原因
- 摘自 维基百科



什么是大数据

- 大数据的定义
 - 大数据指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、高增长率和多样化的信息资产。
- 大数据能做什么？
 - 企业组织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。
 - -----摘自 百度百科



什么是大数据

- 大数据的定义

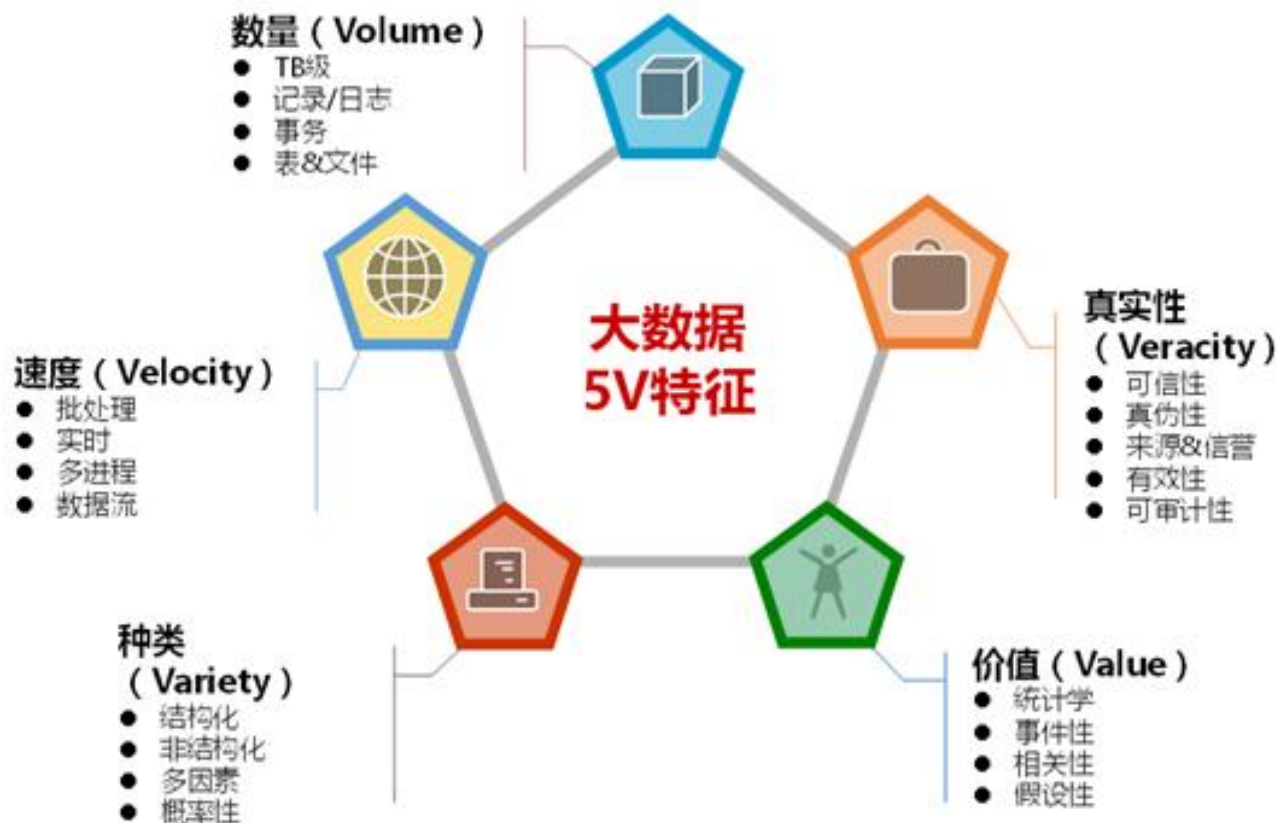
- 大数据是指无法在一定时间内用常规软件工具对其内容进行抓取、管理和处理的数据集合。大数据技术，是指从各种各样类型的数据中，快速获得有价值信息的能力。适用于大数据的技术，包括大规模并行处理数据库，数据挖掘电网，分布式文件系统，分布式数据库，云计算平台，互联网，和可扩展的存储系统。

—

----摘自 MBA智库



大数据的特性



大数据特性

- 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PB、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处理。
 - (V)eracity(准确性)
即处理的结果要保证一定的准确性。
 - (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
 - 高可靠性、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
 - Mapreduce是由Map和reduce组成，Map是影射，把指令分发到多个worker上去，reduce是规约，把Map的worker计算出来的结果合并



Hadoop 起源

- 2003 年开始 google 陆续发表了几篇论文：
 - GFS , MapReduce , BigTable
 - BigTable 存储结构化数据。
 - BigTable 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



Hadoop 起源

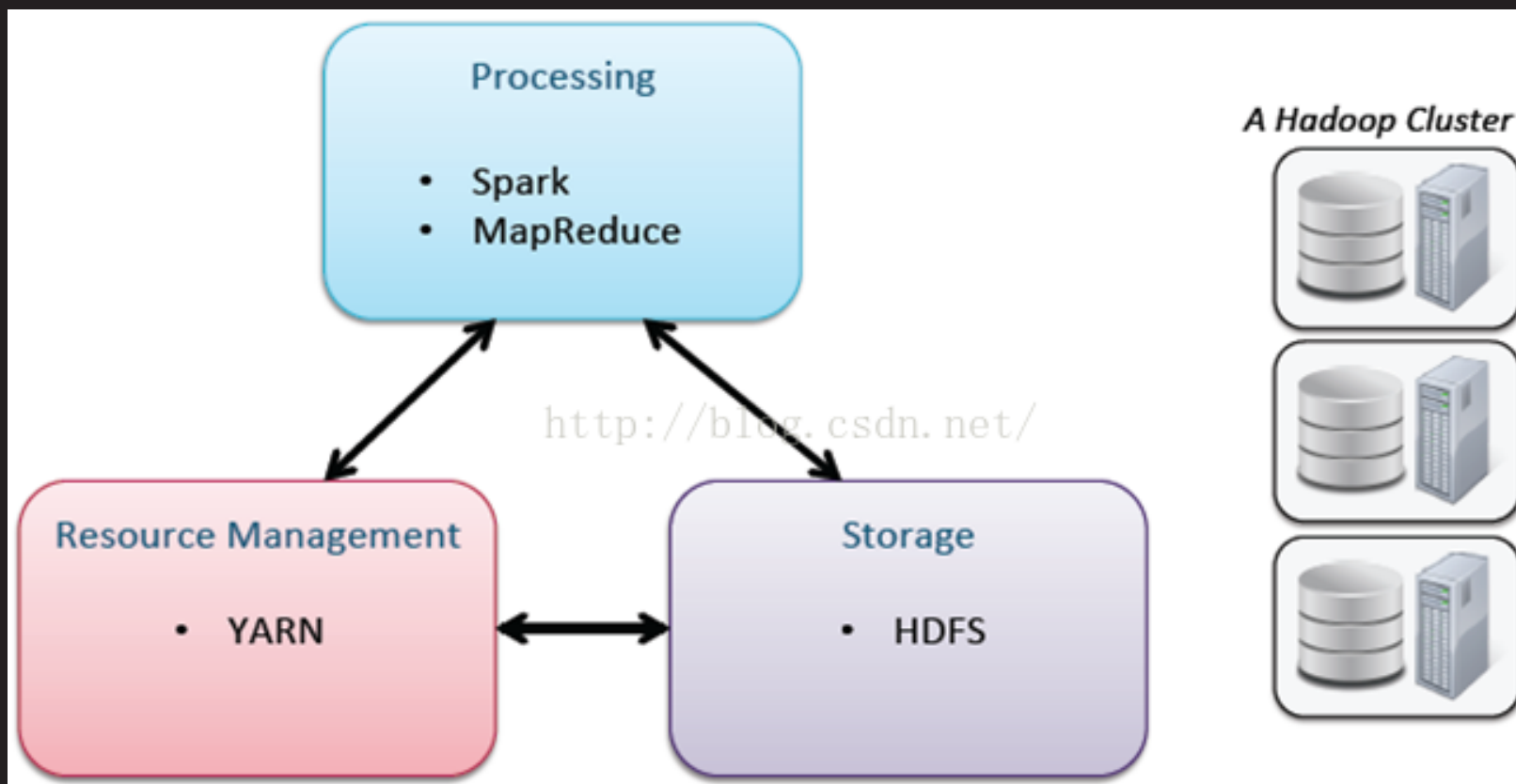
- 这三大技术被称为 Google 的三驾马车。
- 虽然Google没有公布这三个产品的源码，但是他发布了这三个产品的详细设计论文。
- Yahoo 资助的 Hadoop 按照这三篇论文的开源Java实现，不过在性能上 Hadoop 比 Google 要差很多。
 - GFS - - -> HDFS
 - Mapreduce - - -> Mapreduce
 - Bigtable - - -> Hbase



Hadoop 组件



Hadoop 核心组件

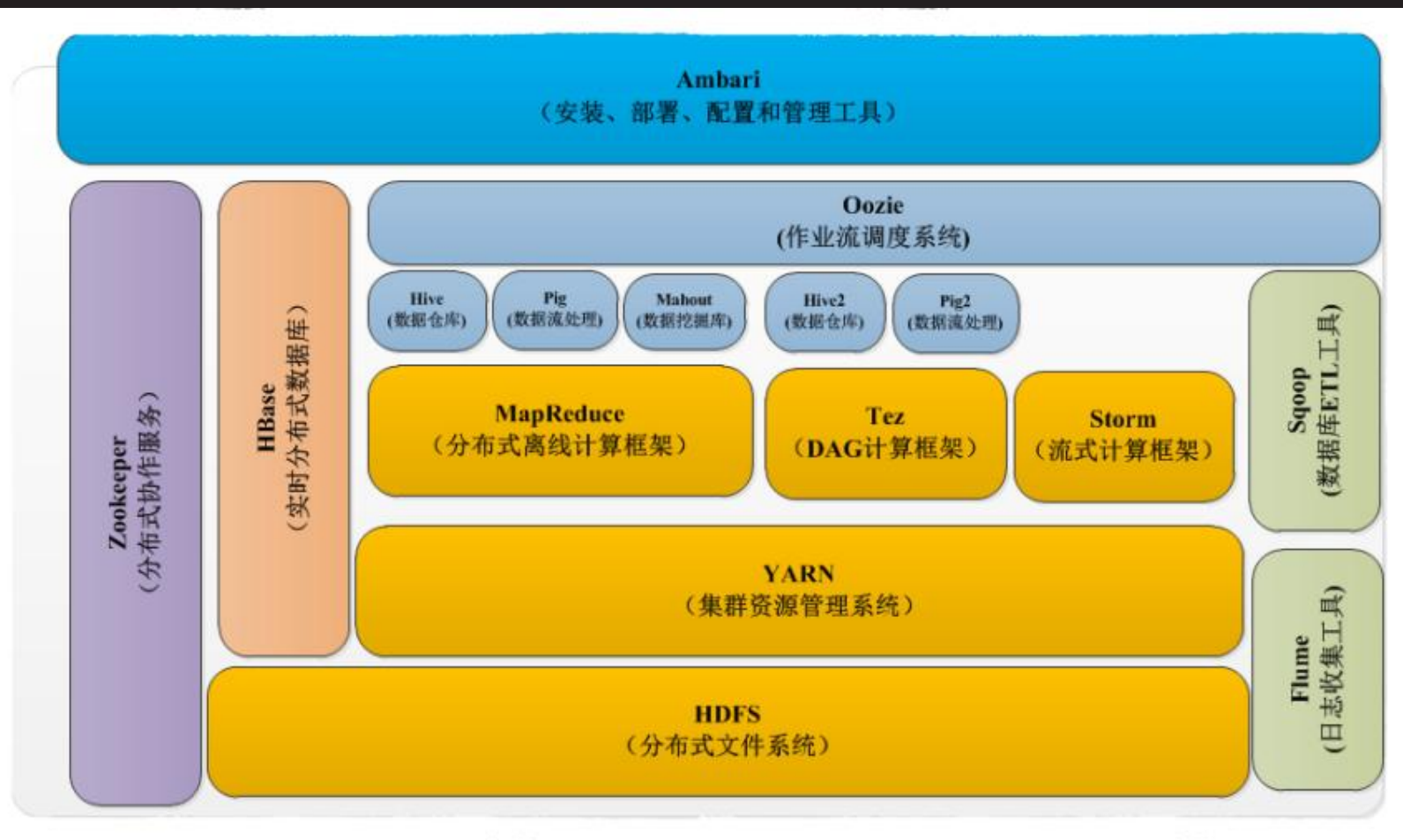


Hadoop 核心组件

- 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 生态系统



Hadoop 常用组件

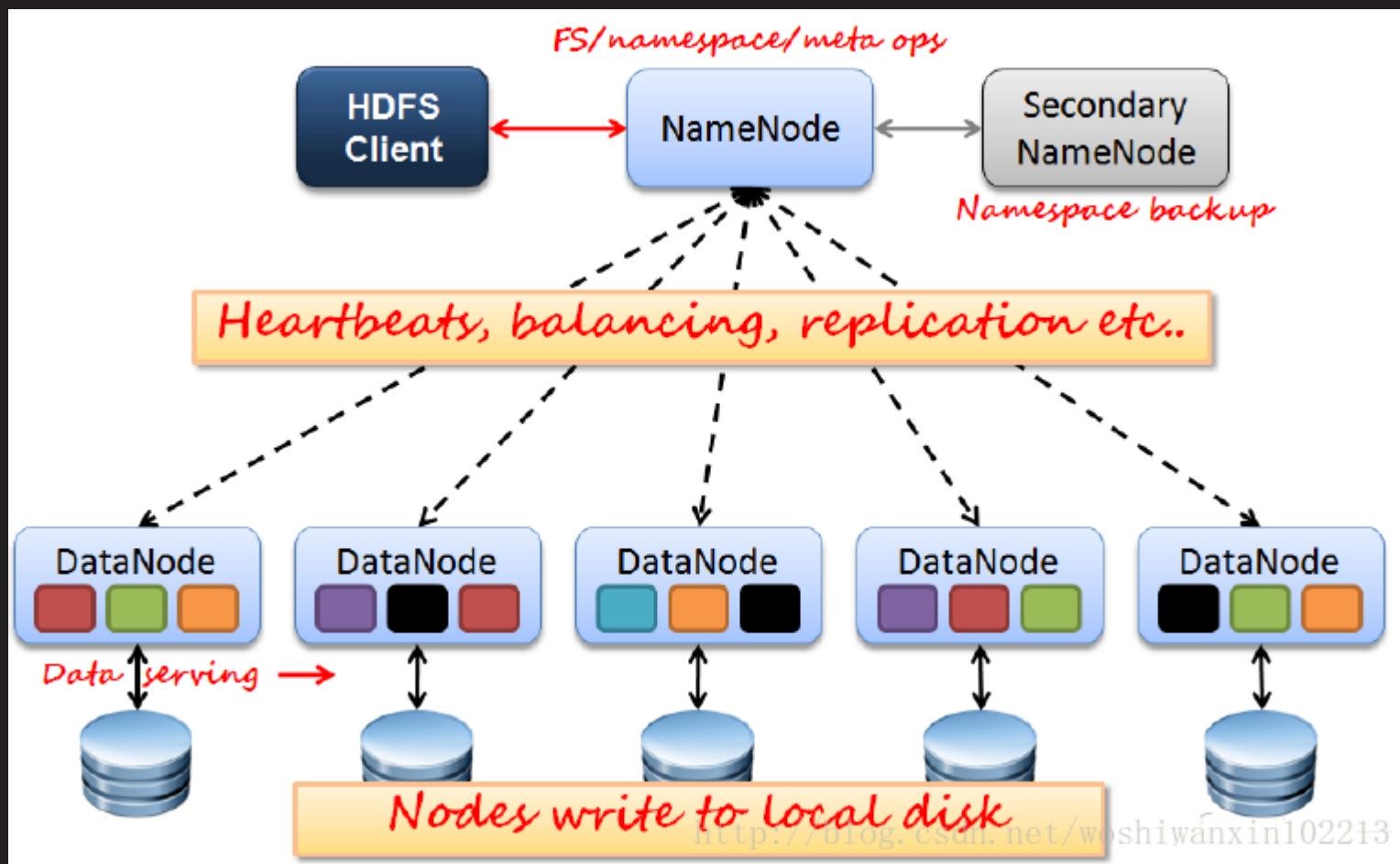
- HDFS (Hadoop分布式文件系统)
- Mapreduce (分布式计算框架)
- Zookeeper (分布式协作服务)
- Hbase (分布式列存数据库)
- Hive (基于Hadoop的数据仓库)
- Sqoop (数据同步工具)
- Pig (基于Hadoop的数据流系统)
- Mahout (数据挖掘算法库)
- Flume (日志收集工具)



Hadoop 核心组件



HDFS 结构



HDFS 角色及概念

- 是Hadoop体系中数据存储管理的基础。它是一个高度容错的系统，用于在低成本的通用硬件上运行。
- 角色和概念
 - Client
 - Namenode
 - Secondarynode
 - Datanode



HDFS 角色及概念

- 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。
- Secondary NameNode
 - 定期合并 fsimage 和fsedits，推送给NameNode
 - 紧急情况下，可辅助恢复NameNode，
- 但Secondary NameNode并非NameNode的热备。



HDFS 角色及概念

- 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。

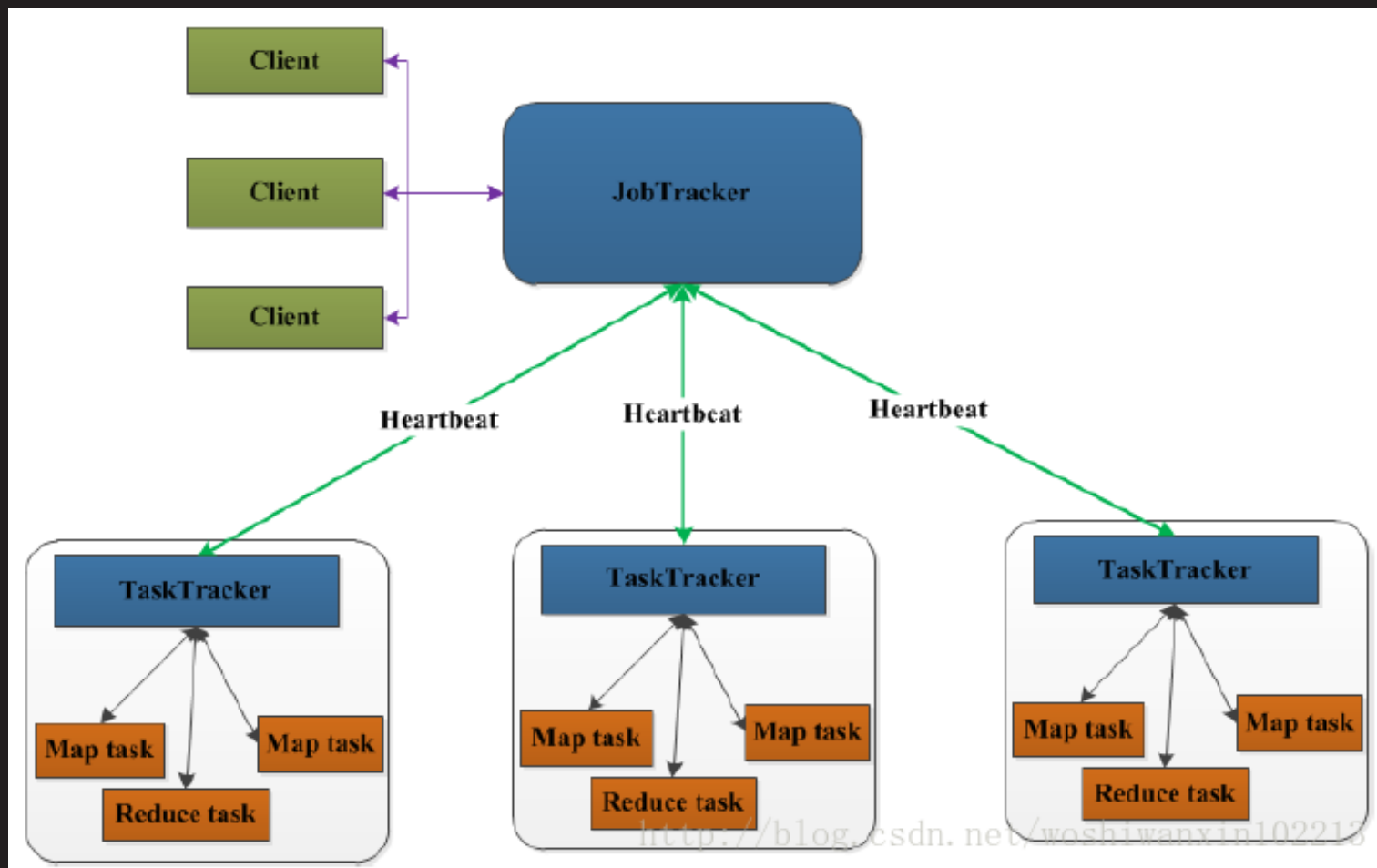


HDFS 角色及概念

- Block
 - 每块缺省64MB大小
 - 每块可以多个副本



MapReduce 结构



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



Mapreduce 角色及概念

- 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。

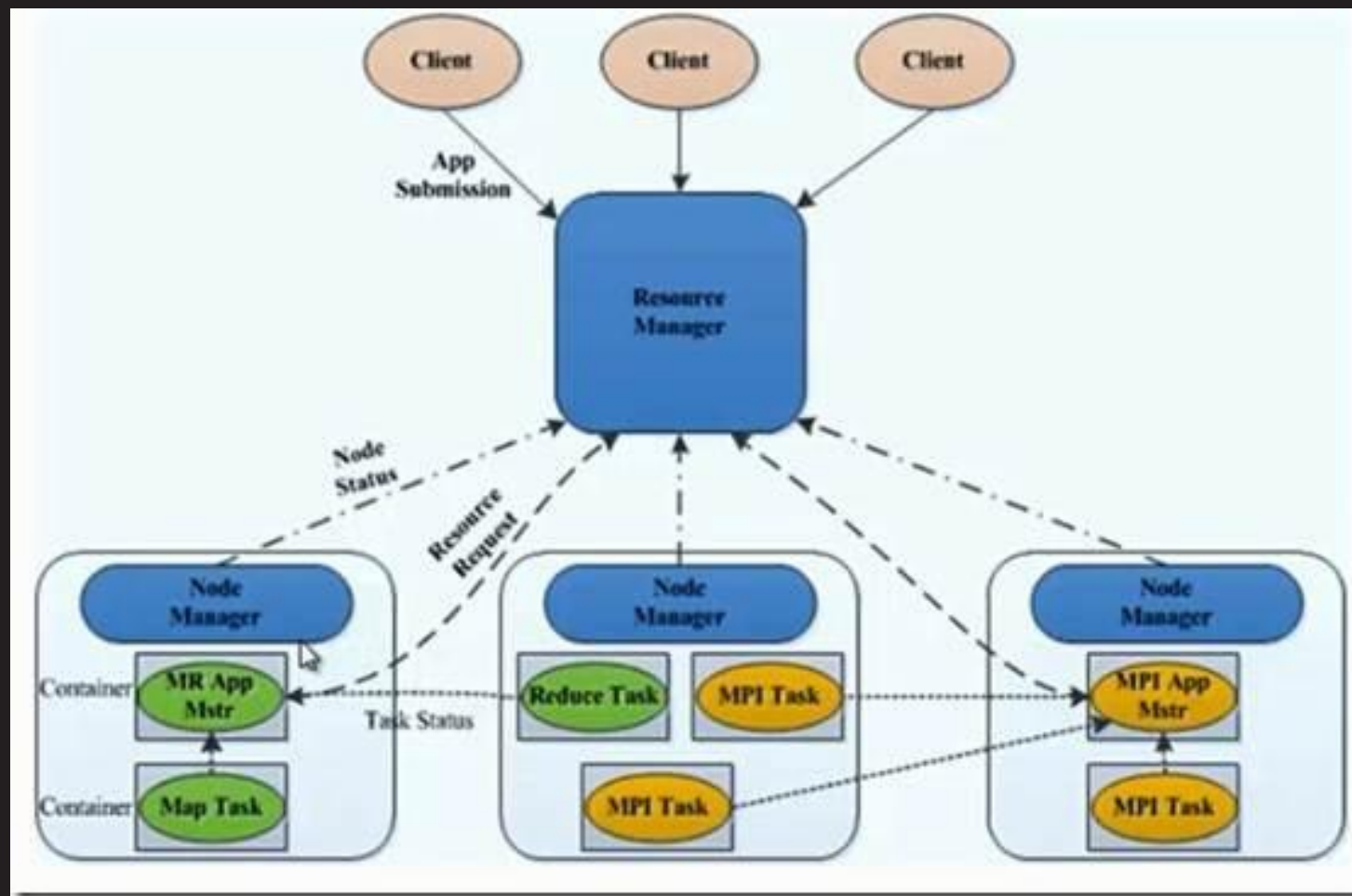


Mapreduce 角色及概念

- 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)。
- Reducer 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。



Yarn 结构



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



Yarn 角色及概念

- 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



Yarn 角色及概念

- Container
 - 对任务运行行环境的抽象，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
 - 任务监控与容错



Yarn 角色及概念

- 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 角色及概念

- 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



Hadoop 安装配置

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>
 - 2、安装配置 java 环境，安装 jps 工具
安装 Openjdk 和 openjdk-devel
 - 3、设置环境变量，启动运行
 - hadoop-env.sh
JAVA_HOME= ""



Hadoop 单机模式安装配置

- 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

```
cd /usr/local/hadoop
mkdir input
cp *.txt input/
./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-
mapreduce-examples-2.7.3.jar wordcount input output
```



Hadoop 伪分布式

- 伪分布式
 - 伪分布式的安装和完全分布式类似，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 配置文件及格式

- 文件格式

- 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>



HDFS 分布式文件系统

完全分布式

- 系统规划

主机	角色	软件
192.168.1.10 Nn01	NameNode SecondaryNameNode	HDFS
192.168.1.11 Node1	DataNode	HDFS
192.168.1.12 Node2	DataNode	HDFS
192.168.1.13 node3	DataNode	HDFS



搭建完全分布式

- 基础环境准备
 - 新开启3台虚拟机
 - 禁用 selinux
`SELINUX=disabled`
 - 禁用 firewalld
`systemctl stop firewalld`
`systemctl mask firewalld`
 - 安装 java-1.8.0-openjdk-devel
`yum install -y java-1.8.0-openjdk-devel`



搭建完全分布式

- 基础环境准备
 - 在3台机器上配置 /etc/hosts
 - 注：所有主机都能 ping 同 namenode 的主机名
 - namenode 能 ping 同所有节点
 - java -version 验证 java 安装
 - jps 验证角色



完全分布式

- 配置 SSH 信任关系 (namenode)
 - 注意：不能出现要求输入 yes 的情况，每台机器都要能登录成功，包括本机！！！！
 - ssh_config
StrictHostKeyChecking no
 - ssh-keygen -b 2048 -t rsa -N "" -f key
 - ssh-copy-id -i ./key.pub root@ip.xx.xx.xx



完全分布式

- 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`



完全分布式

- 环境配置文件 `hadoop-env.sh`
 - openjdk 的安装目录
 - JAVA_HOME
 - hadoop 配置文件的存放目录
 - HADOOP_CONF_DIR



完全分布式

- 核心配置文件 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://nn01:9000</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/var/hadoop</value>
</property>
```



完全分布式

- HDFS 配置文件 hdfs-site.xml
 - namenode 地址声明
 - dfs.namenode.http-address
 - secondarynamenode 地址声明
 - dfs.namenode.secondary.http-address
 - 文件冗余份数
 - dfs.replication



完全分布式

- HDFS 配置文件 hdfs-site.xml

```
<property>
  <name>dfs.namenode.http-address</name>
  <value>nn01:50070</value>
</property>
<property>
  <name>dfs.namenode.secondary.http-
address</name>
  <value>nn01:50090</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>2</value>
</property>
```



完全分布式

- 节点配置文件 slaves
 - 只写 datanode 节点的主机名称
node1
node2
node3
 - 同步配置
 - hadoop 所有节点的配置参数完全一样，我们在一台配置好以后，要把配置文件分发到其它所有主机上去



完全分布式

- HDFS 完全分布式配置
 - 在所有机器上创建 /var/hadoop 文件夹
`mkdir /var/hadoop`
 - 在 namenode 上执行格式化操作
`./bin/hdfs namenode -format`
 - 启动集群
`./sbin/start-dfs.sh`



完全分布式

- JPS验证角色：

- namenode 验证

```
[root@nn01 hadoop]# jps
29826 SecondaryNameNode
31237 Jps
29643 NameNode
```

- datanode 验证

```
[root@node1 ~]# jps
24472 Jps
24027 DataNode
```



完全分布式

- 节点验证：

- namenode 上

- bin/hdfs dfsadmin -report

```
[root@nn01 hadoop]# bin/hdfs dfsadmin -report
Configured Capacity: 51505004544 (47.97 GB)
DFS Used: 733184 (716 KB)
```

```
... ..
```

```
Missing blocks: 0
```

```
Missing blocks (with replication factor 1): 0
```

```
-----
```

```
Live datanodes (3):
```



课程知识点总结

- 大数据的 5V 特性是什么？
 - (V)olume (大体量)
 - (V)ariety(多样性)
 - (V)elocity(时效性)
 - (V)eracity(准确性)
 - (V)alue(大价值)



课程知识点总结

- Hadoop 是用什么语言开发的？
 - JAVA
- Hadoop 的三大核心组件是什么？
 - Hdfs
 - Mapreduce
 - Yarn



课程知识点总结

- Hadoop 有几种部署模式？
 - 单机
 - 伪分布式
 - 完全分布式
- 列举 5 种 Hadoop 的常见组件？



总结答疑
