北风网 IBEIFENG.COM

大数据Hadoop高薪直通车课程

高级 Hadoop 2.x

www.ibeifeng.com

讲师：轩宇（北风网版权所有)

课程大纲

1. 分布式部署Hadoop 2.x
2. 分布式服务框架Zookeeper
3. HDFS HA 架构部署测试
4. HDFS 2.x中高级特性
5. YARN HA架构部署测试

www.ibeifeng.com

课程大纲

1. 分布式部署Hadoop 2.x
2. 分布式服务框架Zookeeper
3. HDFS HA 架构部署测试
4. HDFS 2.x中高级特性
5. YARN HA架构部署测试

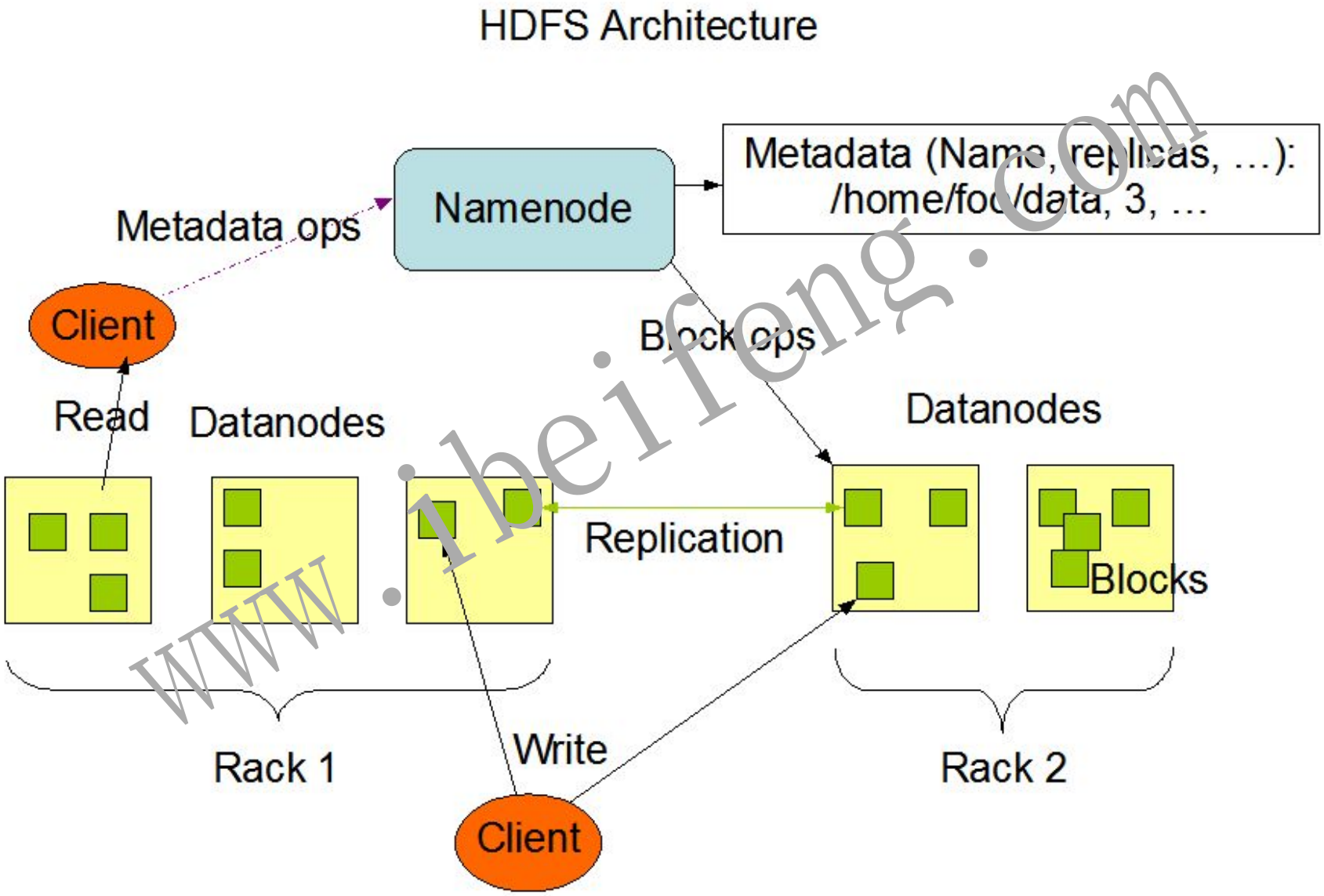
www.ibeifeng.com

分布式集群安装

- 基于伪分布式环境安装进行展开
- 规划机器与服务（★★★★☆）
  - HDFS 文件系统
  - YARN “云操作系统”
  - JobHistoryServer 历史服务监控
- 修改配置文件，设置服务运行机器节点（★★★☆☆）
- 分发HADOOP安装包至各个机器节点
- 依据官方集群安装文档，分别启动各节点相应服务
- 测试 HDFS 、YARN、 MapReduce ， Web UI 监控集群（★★★☆☆）
- 配置主节点至各从节点 SSH 无密钥登陆
- 集群基准测试 （实际环境中必须的）→ **面试题**

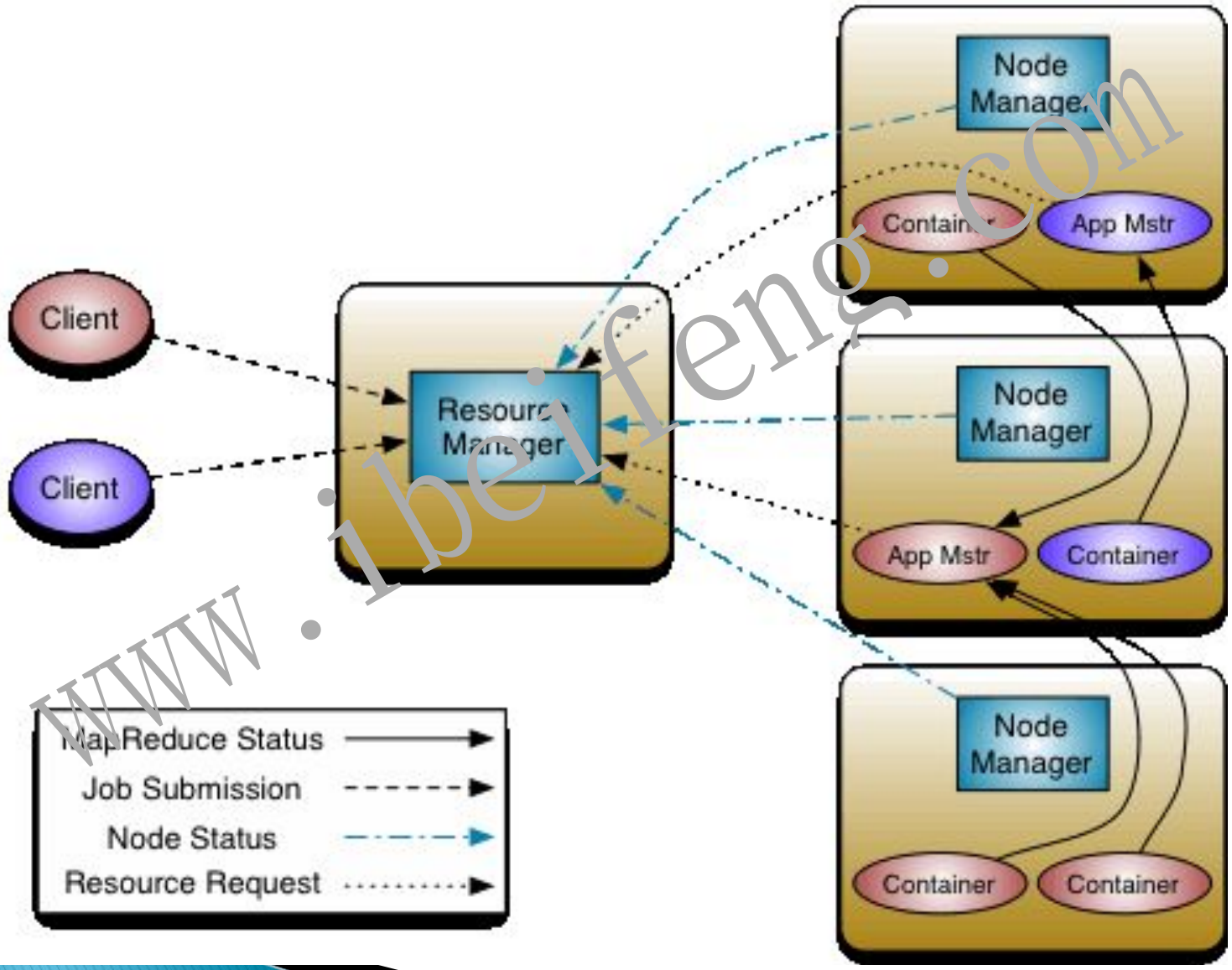
www.ibeifeng.com

HDFS Architecture



www.ibeifeng.com

YARN Architecture



www.ibeifeng.com

课程大纲

1. 分布式部署Hadoop 2.x
2. 分布式服务框架Zookeeper
3. HDFS HA 架构部署测试
4. HDFS 2.x中高级特性
5. YARN HA架构部署测试

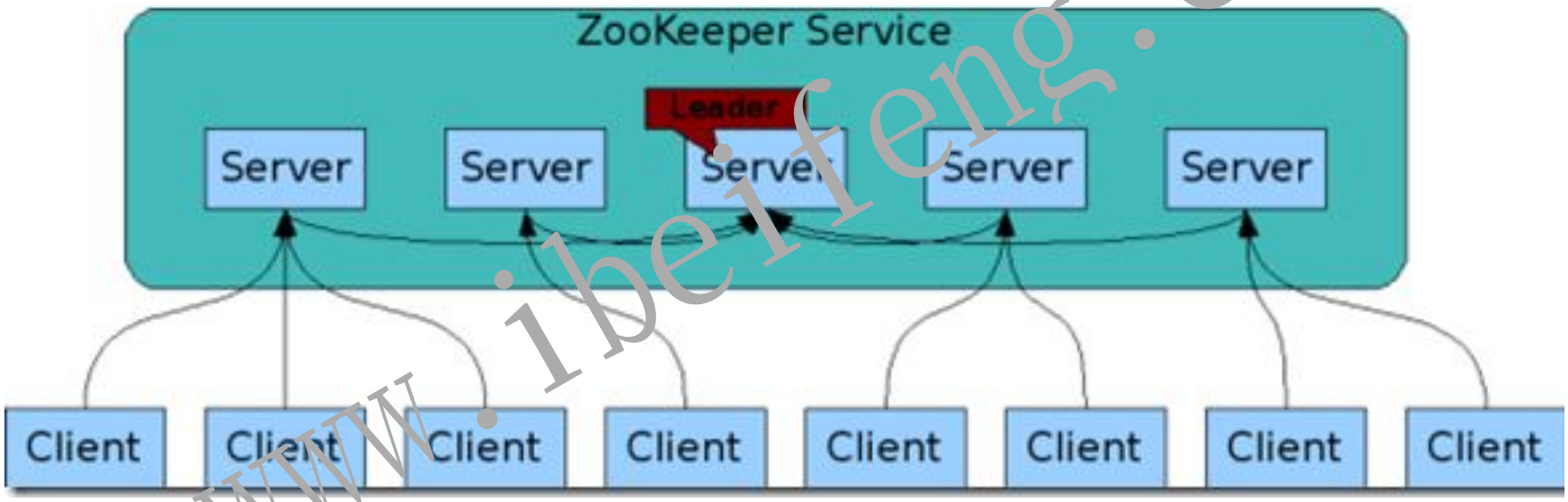
www.ibeifeng.com

What is ZooKeeper?

- 一个开源的分布式的，为分布式应用提供协调服务的Apache项目。
- 提供一个简单的原语集合，以便于分布式应用可以在它之上构建更高层次的同步服务。
- 设计非常易于编程，它使用的是类似于文件系统那样的树形数据结构。
- 目的就是将分布式服务不再需要由于协作冲突而另外实现协作服务。

www.ibeifeng.com

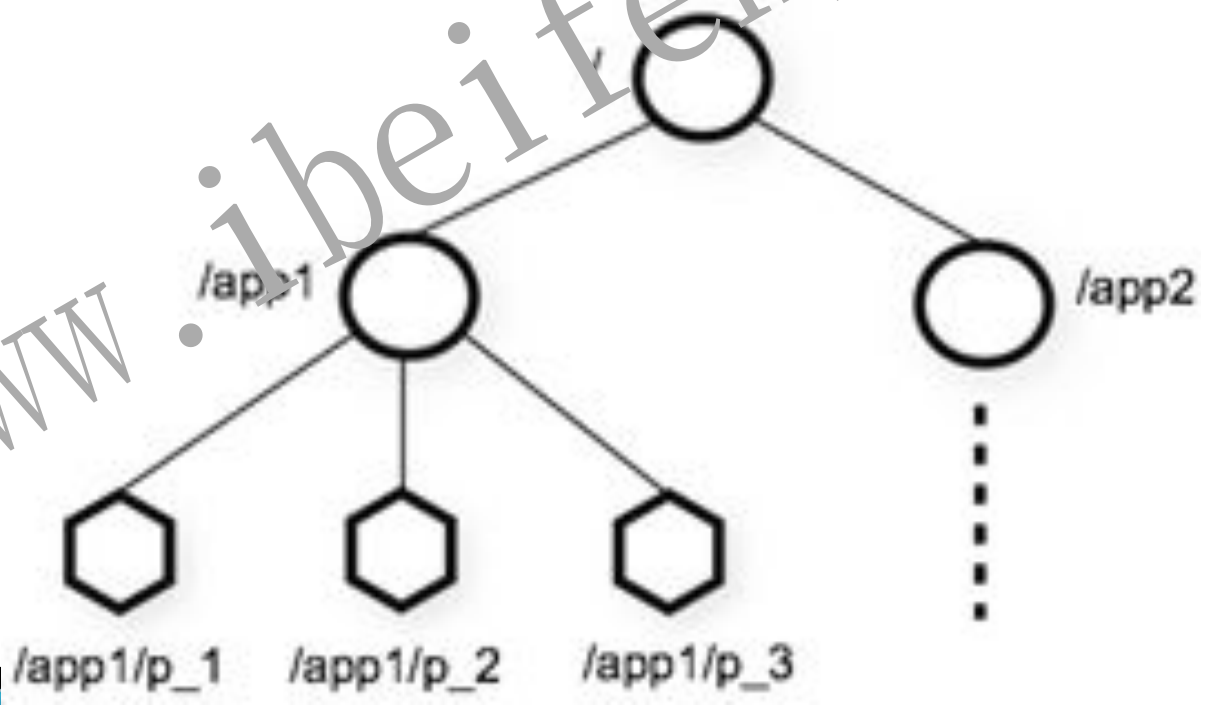
ZooKeeper Service



www.ibeifeng.com

Zookeeper数据结构

- 数据结构和分等级的命名空间
  - Zookeeper的命名空间的结构和文件系统很像。一个名字和文件一样使用/的路径表现，zookeeper的每个节点都是被路径唯一标识。
  - ZooKeeper's Hierarchical Namespace



www.ibeifeng.com

Zookeeper 角色

角色		描述
领导者（Leader）		领导者负责进行投票的发起和决议，更新系统状态
学习者（Learner）	跟随者（Follower）	Follower 用于接收客户请求并向客户端返回结果，在选主过程中参与投票
	观察者（ObServer）	ObServer 可以接收客户端连接，将写请求转发给 leader 节点。但 ObServer 不参加投票过程，只同步 leader 的状态。ObServer 的目的是为了扩展系统，提高读取速度
客户端（Client）		请求发起方

www.ibeifeng.com

ZooKeeper 典型应用场景

- Zookeeper 从设计模式角度来看，是一个**基于观察者模式设计**的分布式服务管理框架，它**负责存储和管理**大家都关心的数据，然后接受**观察者的注册**，一旦这些数据的状态发生变化，Zookeeper 就将负责通知已经在 Zookeeper 上注册的那些**观察者**做出相应的反应，从而实现集群中**类似 Master/Slave 管理模式**。
- 应用场景
  - 统一命名服务（Name Service）
  - 配置管理（Configuration Management）
  - 集群管理（Group Membership）
  - 共享锁（Locks）/同步锁

www.ibeifeng.com

Zookeeper 单机模式安装

- 安装JDK、配置环境变量、验证java –version
- 下载、赋执行权限、解压
  - 下载地址：http://apache.dataguru.cn/zookeeper/
  - 权限：chmod u+x zookeeper-3.4.5.tar.gz
  - 解压：tar -zxvf zookeeper-3.4.5.tar.gz -C /opt/modules/
- 配置
  - 复制配置文件：cp conf/zoo_sample.cfg conf/zoo.cfg
  - 配置数据存储目录：dataDir=/opt/modules/zookeeper-3.4.5/data
  - 创建数据存储目录：mkdir /opt/modules/zookeeper-3.4.5/data
- 启动
  - 启动：bin/zkServer.sh start
- 检测
  - 查看状态：bin/zkServer.sh status
  - Client Shell：bin/zkCli.sh

www.ibeifeng.com

Zookeeper 配置参数详解

- tickTime：这个时间是作为 Zookeeper 服务器之间或客户端与服务器之间维持心跳的时间间隔，也就是每个 tickTime 时间就会发送一个心跳。
- dataDir：顾名思义就是 Zookeeper 保存数据的目录，默认情况下，Zookeeper 将写数据的日志文件也保存在这个目录里。
- clientPort：这个端口就是客户端连接 Zookeeper 服务器的端口，Zookeeper 会监听这个端口，接受客户端的访问请求。
- Zookeeper Client 命令讲解
  - 命令：bin/zkCli.sh -server localhost:2181
  - 详解：ls、get、create、delete、set

www.ibeifeng.com

Zookeeper 分布式安装

- 在单机模式基础之上，修改配置文件conf/zoo.cfg
  - vi conf/zoo.cfg
  - 内容如下：

    initLimit=5

    syncLimit=2

    server.1=192.168.48.128:2888:3888

    server.2=192.168.48.181:2888:3888

    server.3=192.168.48.1822:2888:3888
- 在各个机器安装的数据存储目录下创建myid文件
  - 命令：touch data/myid
  - 编辑：vi data/myid
  - 内容（不同机器，不一样）：

    192.168.48.128：1

    192.168.48.128：2

    192.168.48.128：3

www.ibeifeng.com

Zookeeper 配置参数详解续

- initLimit：这个配置项是用来配置 Zookeeper 接受客户端（这里所说的客户端不是用户连接 Zookeeper 服务器的客户端，而是 Zookeeper 服务器集群中连接到 Leader 的 Follower 服务器）初始化连接时最长能忍受多少个心跳时间间隔数。当已经超过 10 个心跳的时间（也就是 tickTime）长度后 Zookeeper 服务器还没有收到客户端的返回信息，那么表明这个客户端连接失败。总的时间长度就是 5*2000=10 秒。
- syncLimit：这个配置项标识 Leader 与 Follower 之间发送消息，请求和应答时间长度，最长不能超过多少个 tickTime 的时间长度，总的时间长度就是 2*2000=4 秒。

www.ibeifeng.com

Zookeeper 配置参数详解续

- **server.A=B:C:D**：其中 A 是一个数字，表示这个是第几号服务器；B 是这个服务器的 ip 地址；C 表示的是这个服务器与集群中的 Leader 服务器交换信息的端口；D 表示的是万一集群中的 Leader 服务器挂了，需要一个端口来重新进行选举，选出一个新的 Leader，而这个端口就是用来执行选举时服务器相互通信的端口。如果是伪集群的配置方式，由于 B 都是一样，所以不同的 Zookeeper 实例通信端口号不能一样，所以要给它们分配不同的端口号。
- 集群模式下配置一个文件 myid，这个文件在 dataDir 目录下，这个文件里面就有一个数据就是 A 的值，Zookeeper 启动时读取此文件，拿到里面的数据与 zoo.cfg 里面的配置信息比较从而判断到底是那个 server。

www.ibeifeng.com

Zookeeper 访问方式-Shell命令

www.1beifeng.com

课程大纲

1. 分布式部署Hadoop 2.x
2. 分布式服务框架Zookeeper
3. HDFS HA 架构部署测试
4. HDFS 2.x中高级特性
5. YARN HA架构部署测试

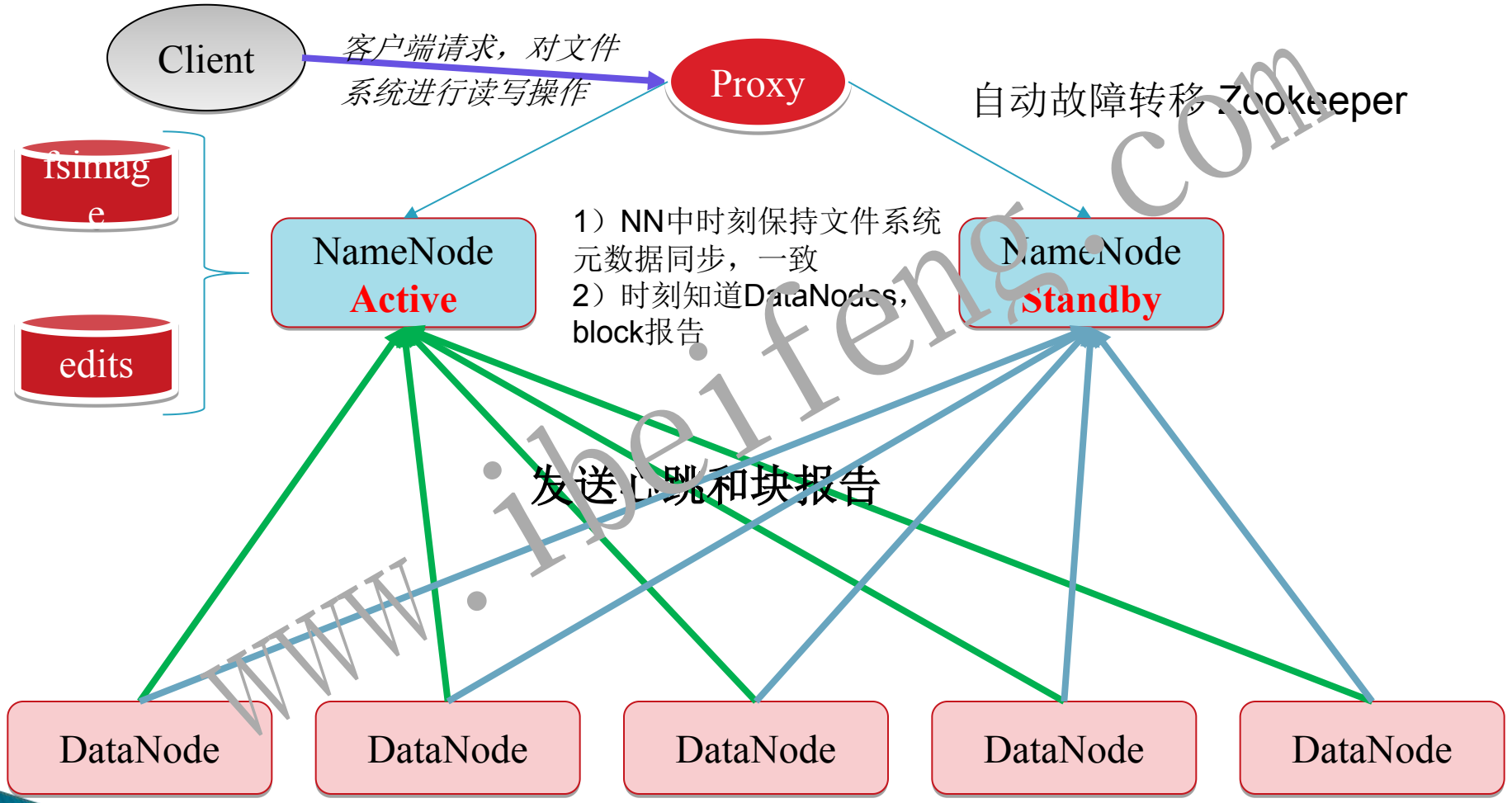
www.ibeifeng.com

背景

- Hadoop 2.0 之前，在HDFS 集群中 NameNode 存在单点故障（SPOF ）。对于只有一个NameNode 的集群，若NameNode 机器出现故障，则整个集群将无法使用，直到NameNode 重新启动。
- NameNode 主要在以下两个方面影响HDFS 集群
  - NameNode 机器发生意外，如宕机，集群将无法使用，直到管理员重启
  - NameNode 机器需要升级，包括软件、硬件升级，此时集群也将无法使用
- HDFS HA 功能通过**配置Active/Standby 两个NameNodes 实现在集群中对NameNode 的热备**来解决上述问题。如果出现故障，如机器崩溃或机器需要升级维护，这时可通过此种方式将NameNode 很快的切换到另外一台机器。

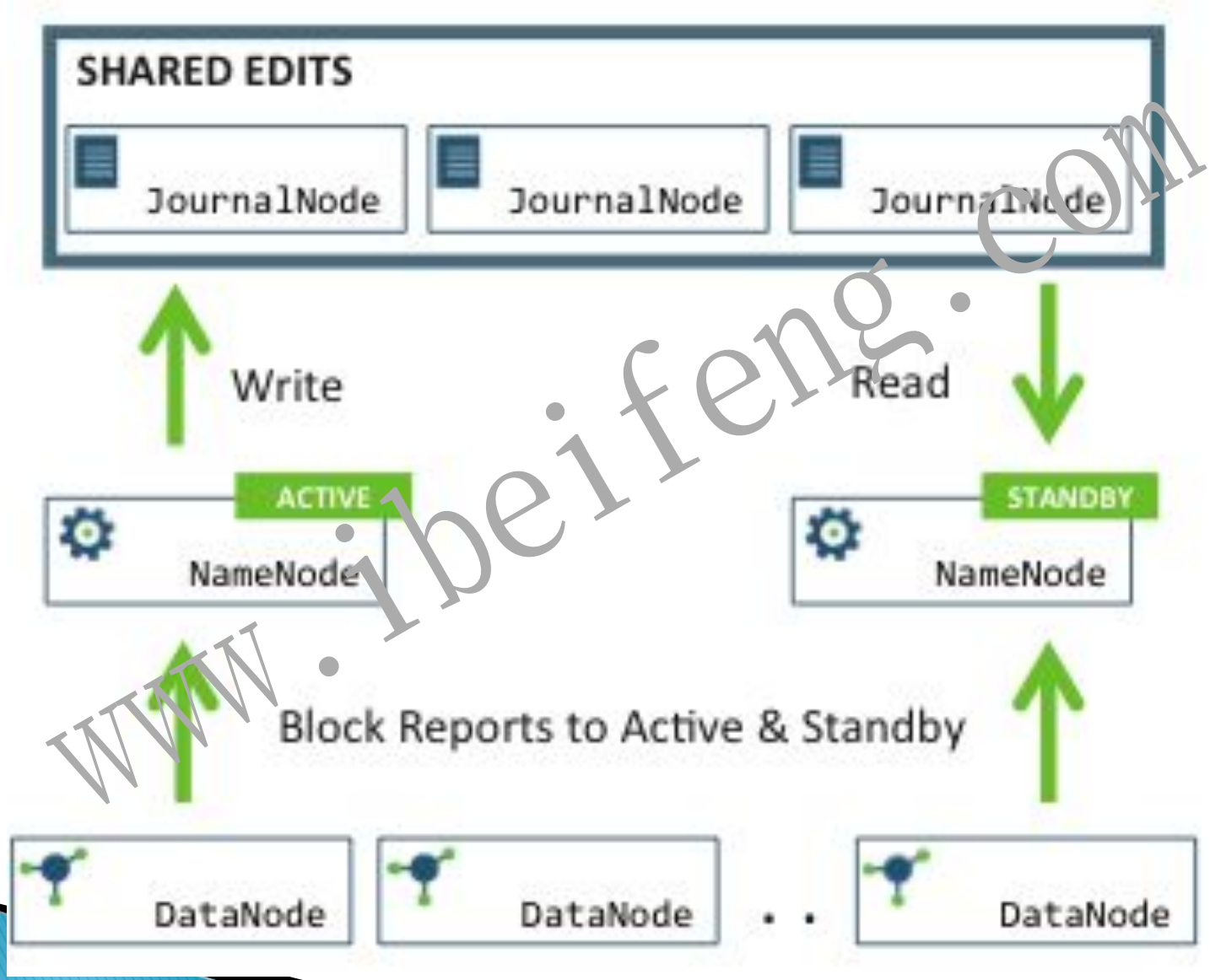
www.ibeifeng.com

HDFS HA 设计



www.ibeifeng.com

HDFS HA 设计



www.ibeifeng.com

QJM HA 配置

- NameNode HA 基本配置（core-site.xml、hdfs-site.xml）
  - Active NameNode与Standby NameNode 地址配置
  - NameNode 与DataNode 本地存储路径配置
  - HDFS Namespace 访问配置
  - 隔离fencing配置（配置两节点之间的互相SSH无密钥登录）
- QJM 配置（hdfs-site.xml）
  - QJM 管理编辑日志
  - 编辑日志存储目录
- 手动故障转移（无需配置）

www.1beifeng.com

QJM HA 启动

- Step1 :在各个JournalNode节点上，输入以下命令启动journalnode服务：

  $ sbin/hadoop-daemon.sh start journalnode

- Step2:在[nn1]上，对其进行格式化，并启动：

  $ bin/hdfs namenode -format

  $ sbin/hadoop-daemon.sh start namenode

- Step3:在[nn2]上，同步nn1的元数据信息：

  $ bin/hdfs namenode -bootstrapStandby

- Step4:启动[nn2]：

  $ sbin/hadoop-daemon.sh start namenode

- Step5:将[nn1]切换为Active

  $ bin/hdfs haadmin -transitionToActive nn1

- Step6:在[nn1]上，启动所有datanode

  $ sbin/hadoop-daemons.sh start datanode

www.ibeifeng.com

NameNode 管理命令

- **bin/hdfs namenode**

  [-backup]

  [-checkpoint]

  [-importCheckpoint]

  [-format [-clusterid cid] [-force] [-nonInteractive]]

  [-upgrade]

  [-rollback]

  [-finalize]

  [-initializeSharedEdits]

  [-bootstrapStandby]

  [-recover [-force]]

www.ibeifeng.com

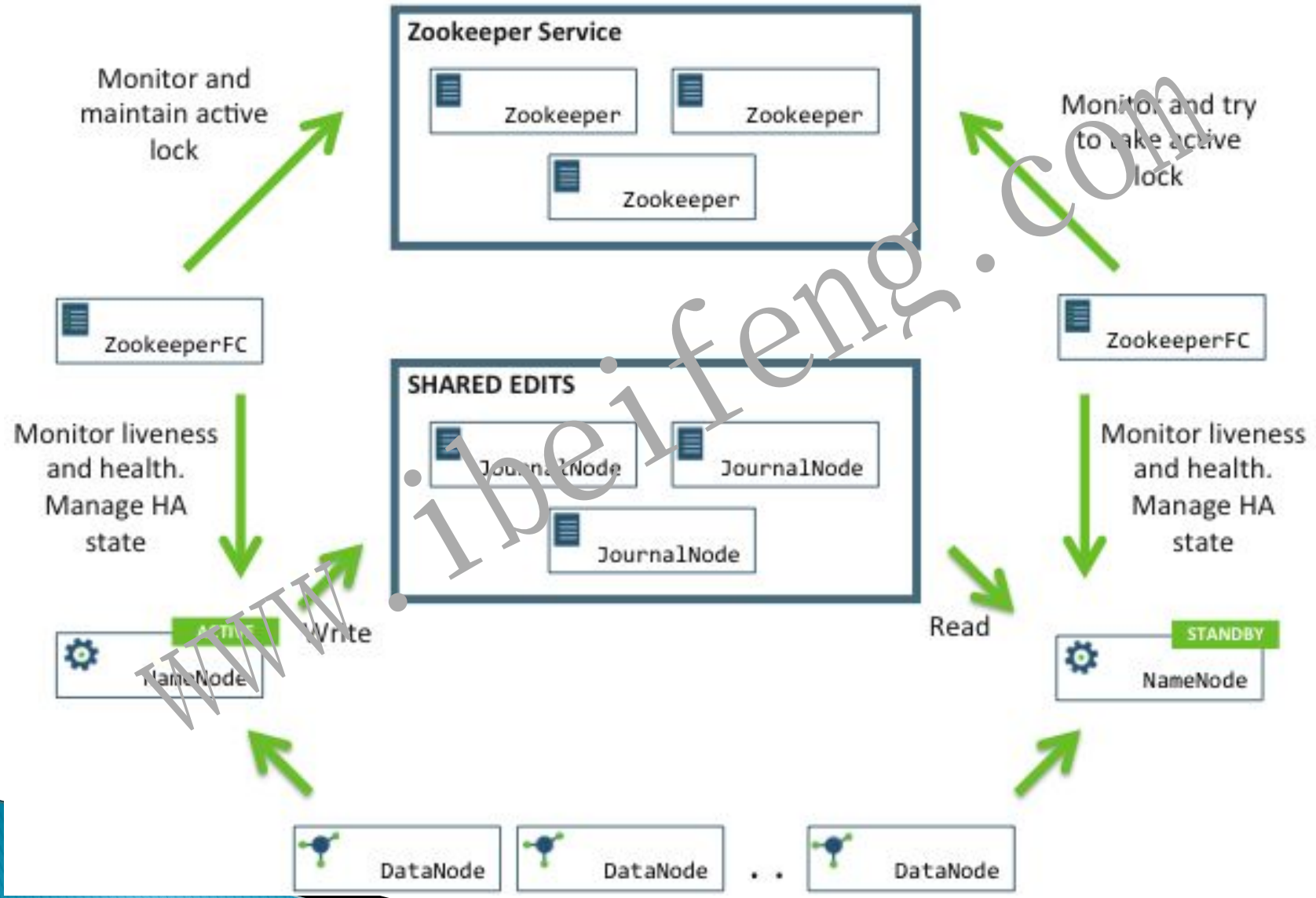
NN HA 自动故障转移

- 启动
  - 关闭所有HDFS 服务 sbin/stop-dfs.sh
  - 启动Zookeeper 集群 bin/zkServer.sh start
  - 初始化 HA 在Zookeeper中状态 bin/hdfs zkfc –formatZK
  - 启动HDFS服务 sbin/start-dfs.sh
  - 在各个NameNode节点上启动DFSZK Failover Controller，先在那台机器启动，那个机器的NameNode就是Active NameNode

    sbin/hadoop-daemin.sh start zkfc
- 验证
  - 将Active NemeNode 进程kill ， 命令：kill -9 pid
  - 将Active NameNode 机器断开网络， 命令：service network stop

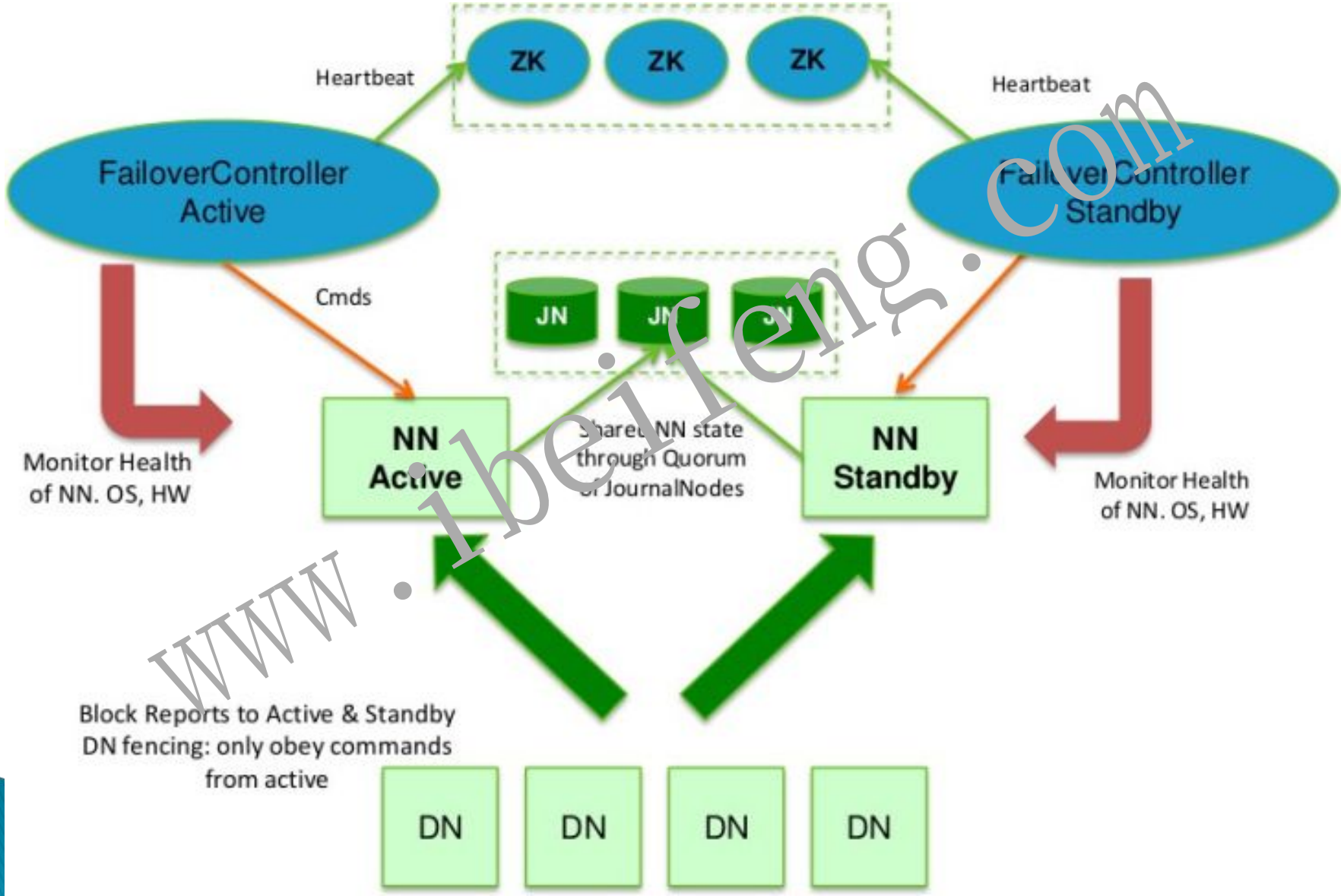
www.beifeng.com

NN HA 自动故障转移



www.ibeifeng.com

HDFS Using QJM



www.ibeifeng.com

课程大纲

1. 分布式部署Hadoop 2.x
2. 分布式服务框架Zookeeper
3. HDFS HA 架构部署测试
4. HDFS 2.x中高级特性
5. YARN HA架构部署测试

www.ibeifeng.com

单 NameNode架构的局限性

- **Namespace（命名空间）的限制**

  由于Namenode在内存中存储所有的元数据（metadata），因此单个Namenode所能存储的对象（文件+块）数目受到Namenode所在JVM的heap size的限制。50G的heap能够存储20亿（200 million）个对象，这20亿个对象支持4000个datanode，12PE的存储（假设文件平均大小为40MB）。随着数据的飞速增长，存储的需求也随之增长。单个datanode从4T增长到36T，集群的尺寸增长到8000个datanode。存储的需求从12PB增长到大于100PB。

- **隔离问题**

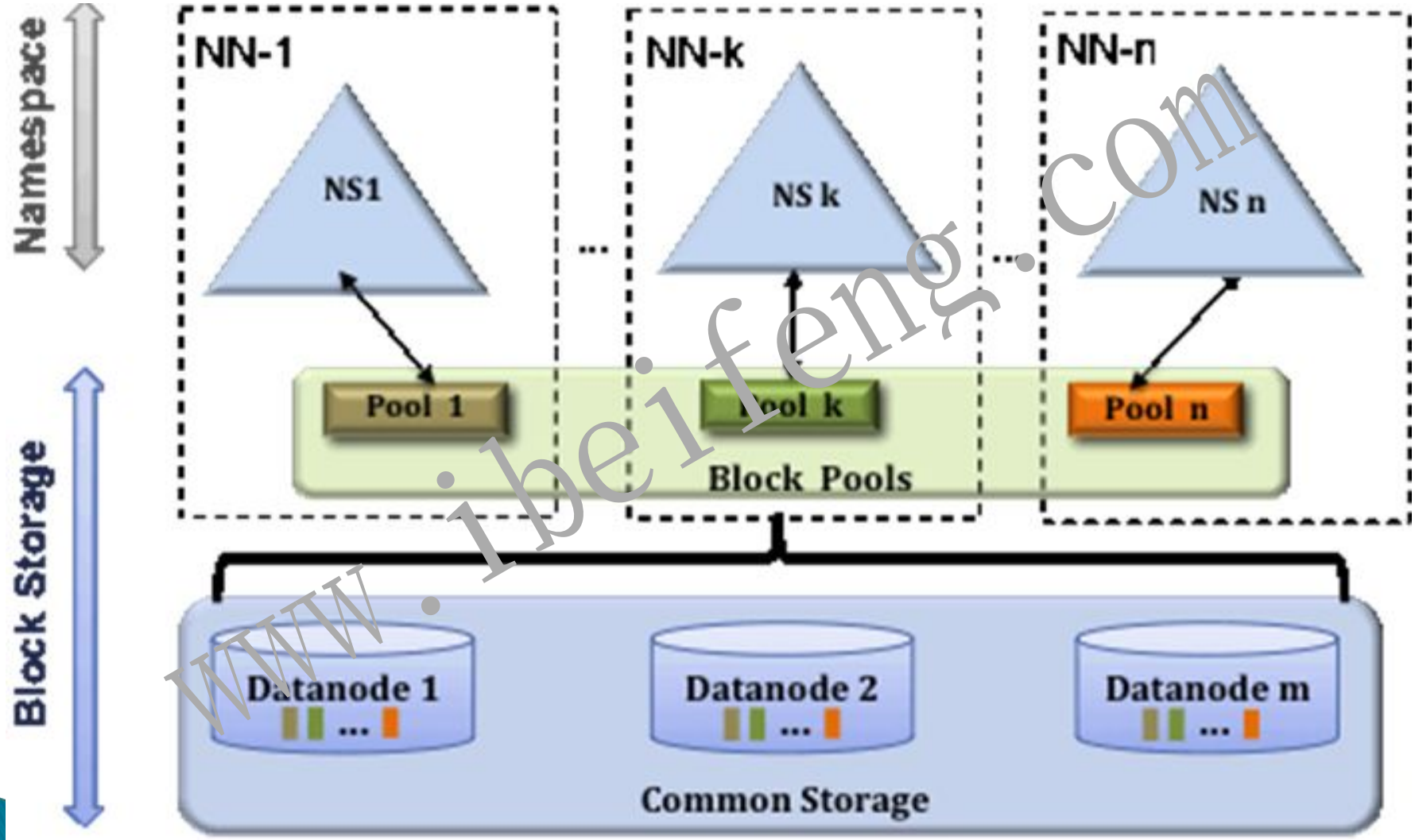
  由于HDFS仅有一个Namenode，无法隔离各个程序，因此HDFS上的一个实验程序就很有可能影响整个HDFS上运行的程序。

- **性能的瓶颈**

  由于是单个Namenode的HDFS架构，因此整个HDFS文件系统的吞吐量受限于单个Namenode的吞吐量。

www.ibeifeng.com

HDFS Federation 架构设计



www.ibeifeng.com

HDFS Federation 应用思考

- 不同应用可以使用不同NameNode 进行数据管理
  - 图片业务、爬虫业务、日志审计业务
  - Hadoop 生态系统中，不同的框架使用不同的NameNode进行管理 Namespace。（隔离性）

www.ibeifeng.com

File System Snapshots

- HDFS快照是一个只读的基于时间点文件系统拷贝。快照可以是整个文件系统的也可以是一部分。常用来作为数据备份，防止用户错误和容灾快照功能。
- HDFS实现功能：
  - Snapshot 创建的时间 复杂度为O(1)，但是不包括INode 的寻找时间
  - 只有当修改SnapShot时，才会有额外的内存占用，内存使用量为O(M),M 为修改的文件或者目录数
  - 在DataNode上面的blocks 不会复制，做Snapshot 的文件是纪录了block的列表和文件的大小，但是没有数据的复制
  - Snapshot 并不会影响HDFS 的正常操作：修改会按照时间的反序记录，这样可以直接读取到最新的数据。快照数据是当前数据减去修改的部分计算出来的。

www.ibeifeng.com

File System Snapshots

- 设置一个目录为可快照

  $ bin/hdfs **dfsadmin** -allowSnapshot <path>

- 取消目录可快照

  $ bin/hdfs **dfsadmin** -disallowSnapshot <path>

- 生成快照

  $ bin/hdfs **dfs -createSnapshot** <path> [<snapshotName>]

- 删除快照

  $ bin/hdfs **dfs -deleteSnapshot** <path> <snapshotName>

- 列出所有可快照目录

  $ bin/hdfs **lsSnapshottableDir**

- 比较快照之间的差异

  $ bin/hdfs **snapshotDiff** <path> <fromSnapshot> <toSnapshot>

www.ibeifeng.com

集中式缓存管理

- Hadoop从2.3.0版本开始支持HDFS缓存机制，HDFS允许用户将一部分目录或文件缓存在HDFS当中，NameNode会通知拥有对应块的DataNodes将其缓存在DataNode的内存当中。
- 优势
  - 防止那些被频繁使用的数据从内存中清除
  - 因为DataNode的缓存由NameNode来管理，applications在做任务安排时可以查询这个缓存的列表，使用一个被缓存的块副本能够提高读性能
  - 当块被DataNode缓存之后，客户端可以使用一个新的、高效的、zero-copy的读API，因为缓存中的数据已经被计算过checksum，当使用新API时，客户端基本上是零开销的
  - 可以提高集群的内存利用率。当使用操作系统的缓存时，对一个块的重复读会导致所有的副本都会被放到缓冲区当中，当使用集中式缓存时，用户可以指定n个副本中的m个才会被缓存，可以节约n-m的内存

www.ibeifeng.com

集中式缓存管理

- 使用场景
  - 集中式缓存对那些频繁访问的的文件是非常有用的，例如hive中经常被使用的fact表就非常适合缓存
  - 另一方面，缓存一年的查询结果可能没那么有用了，因为这个结果可能只会被查看一次
  - 有助于提高混合类型作业的SLA性能，把高优先级的数据缓存起来可以确保它不会与低优先级的数据竞争磁盘IO

www.ibeifeng.com

cacheadmin command-line interface

```
$ bin/hdfs cacheadmin -listPools
Found 0 results.

$ bin/hdfs cacheadmin -addPool conf-pool
Successfully added cache pool conf-pool.

$ bin/hdfs cacheadmin -listPools
Found 1 result.
NAME       OWNER   GROUP   MODE           LIMIT  MAXTTL
conf-pool  hadoop  hadoop  rwxr-xr-x   unlimited   never

$ bin/hdfs cacheadmin -addDirective -path /user/hadoop/conf -pool conf-pool
Added cache directive 1

$ bin/hdfs cacheadmin -listDirectives -stats
Found 1 entry
 ID POOL      REPL EXPIRY  PATH                BYTES_NEEDED  BYTES_CACHED
  1 conf-pool    1 never   /user/hadoop/conf           1923             0

$ bin/hdfs cacheadmin -removeDirective 1
Removed cached directive 1
$ bin/hdfs cacheadmin -listDirectives -stats
Found 0 entries
```

1. 查看Pools
2. 创建一个Pool
3. 创建一个缓存目录或者文件
4. 查看某个pool下所有的缓存对象
5. 依据缓存对象ID进行删除

www.ibeifeng.com

Distributed Copy

- **DistCp Version 2 (distributed copy) is a tool used for large inter/intra-cluster copying.** It uses MapReduce to effect its distribution, error handling and recovery, and reporting. It expands a list of files and directories into input to map tasks, each of which will copy a partition of the files specified in the source list.
- The erstwhile implementation of DistCp has its share of quirks and drawbacks, both in its usage, as well as its extensibility and performance. **The purpose of the DistCp refactor was to fix these shortcomings, enabling it to be used and extended programmatically.** New paradigms have been introduced to improve runtime and setup performance, while simultaneously retaining the legacy behaviour as default.

www.ibeifeng.com

HFTP Introduction

- HFTP is a Hadoop filesystem implementation that lets you read data from a remote Hadoop HDFS cluster. The reads are done via HTTP, and data is sourced from DataNodes.
- HFTP is a read-only filesystem, and will throw exceptions if you try to use it to write data or modify the filesystem state.
- HFTP is primarily useful if you have multiple HDFS clusters with different versions and you need to move data from one to another. HFTP is wire-compatible even between different versions of HDFS.

www.ibeifeng.com

课程大纲

分布式部署Hadoop 2.x

分布式服务框架Zookeeper

HDFS HA 架构部署测试

HDFS 2.x中高级特性

YARN HA架构部署测试

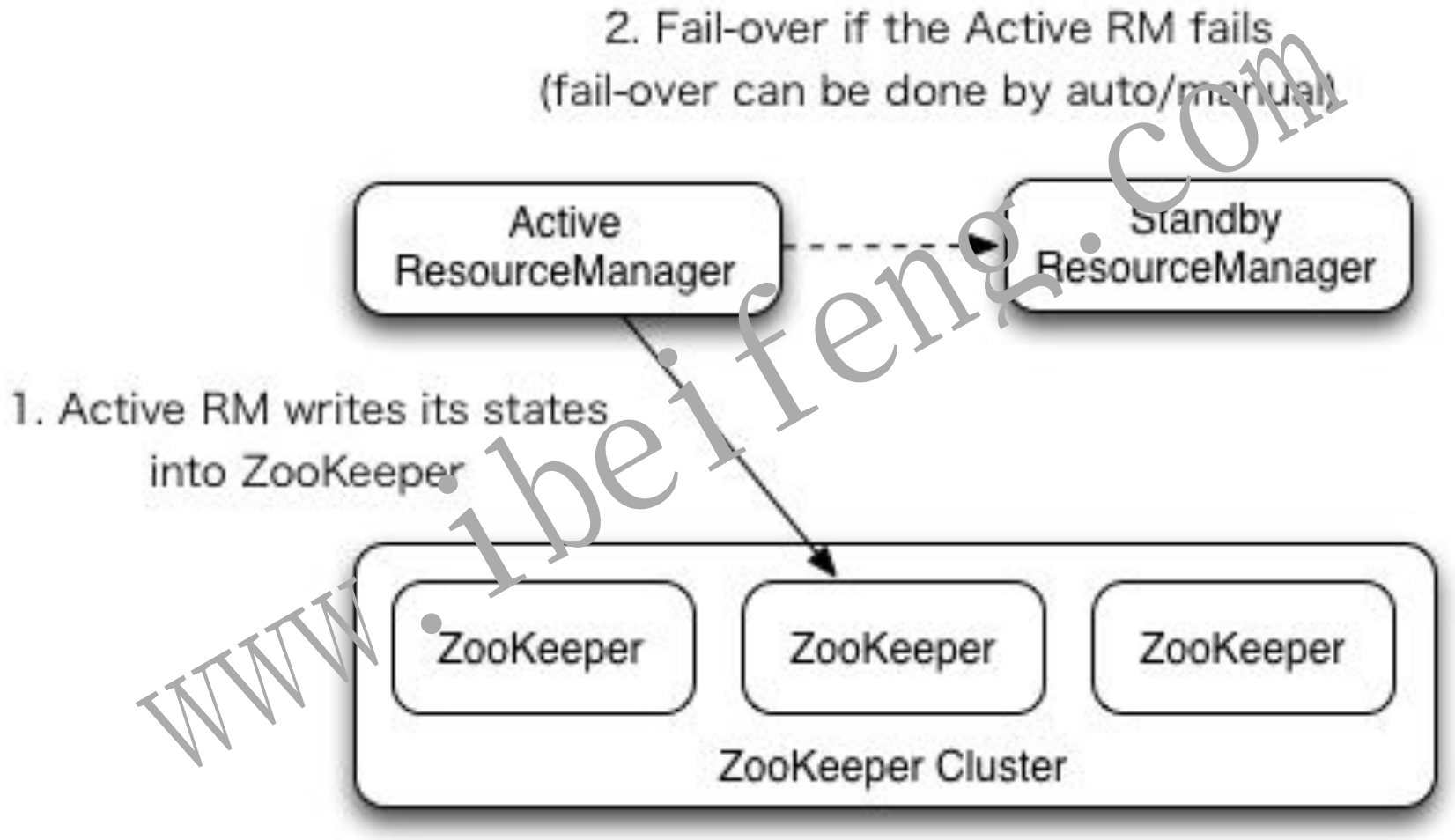
www.ibeifeng.com

ResourceManager High Availability

- The ResourceManager (RM) is responsible for tracking the resources in a cluster, and scheduling applications (e.g., MapReduce jobs).
- Prior to Hadoop 2.4, the ResourceManager is the single point of failure in a YARN cluster.
- The High Availability feature adds redundancy in the form of an Active/Standby ResourceManager pair to remove this otherwise single point of failure.

www.ibeifeng.com

ResourceManager High Availability



www.ibeifeng.com

ResourceManager High Availability

Configuration Property	Description
yarn.resourcemanager.zk-address	Address of the ZK-quorum. Used both for the state-store and embedded leader-election.
yarn.resourcemanager.ha.enabled	Enable RM HA
yarn.resourcemanager.ha.rm-ids	List of logical IDs for the RMs. e.g., "rm1,rm2"
yarn.resourcemanager.hostname.*rm-id*	For each *rm-id*, specify the hostname the RM corresponds to. Alternately, one could set each of the RM's service addresses.
yarn.resourcemanager.ha.id	Identifies the RM in the ensemble. This is optional; however, if set, admins have to ensure that all the RMs have their own IDs in the config
yarn.resourcemanager.ha.automatic-failover.enabled	Enable automatic failover; By default, it is enabled only when HA is enabled.
yarn.resourcemanager.ha.automatic-failover.embedded	Use embedded leader-elector to pick the Active RM, when automatic failover is enabled. By default, it is enabled only when HA is enabled.
yarn.resourcemanager.cluster-id	Identifies the cluster. Used by the elector to ensure an RM doesn't take over as Active for another cluster.
yarn.client.failover-proxy-provider	The class to be used by Clients, AMs and NMs to failover to the Active RM.
yarn.client.failover-max-attempts	The max number of times FailoverProxyProvider should attempt failover.
yarn.client.failover-sleep-base-ms	The sleep base (in milliseconds) to be used for calculating the exponential delay between failovers.
yarn.client.failover-sleep-max-ms	The maximum sleep time (in milliseconds) between failovers
yarn.client.failover-retries	The number of retries per attempt to connect to a ResourceManager.
yarn.client.failover-retries-on-socket-timeouts	The number of retries per attempt to connect to a ResourceManager on socket timeouts.

www.ibelieng.com

ResourceManager High Availability

Sample configurations

Here is the sample of minimal setup for RM failover.

```
<property>
  <name>yarn.resourcemanager.ha.enabled</name>
  <value>true</value>
</property>
<property>
  <name>yarn.resourcemanager.cluster-id</name>
  <value>cluster1</value>
</property>
<property>
  <name>yarn.resourcemanager.ha.rm-ids</name>
  <value>rm1,rm2</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname.rm1</name>
  <value>master1</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname.rm2</name>
  <value>master2</value>
</property>
<property>
  <name>yarn.resourcemanager.zk-address</name>
  <value>zk1:2181,zk2:2181,zk3:2181</value>
</property>
```

www.1beifeng.com

ResourceManager High Availability

Admin commands

yarn rmadmin has a few HA-specific command options to check the health/state of an RM, and transition to Active/Standby. Commands for HA take service id of RM set by yarn.resourcemanager.ha.rm-ids as argument.

```
$ yarn rmadmin -getServiceState rm1
active

$ yarn rmadmin -getServiceState rm2
standby
```

If automatic failover is enabled, you can not use manual transition command.

```
$ yarn rmadmin -transitionToStandby rm1
Automatic failover is enabled for org.apache.hadoop.yarn.client.RMHAServiceTarget@1d8299fd
Refusing to manually manage HA state, since it may cause
a split-brain scenario or other incorrect state.
If you are very sure you know what you are doing, please
specify the forcemanual flag.
```

www.ibeifeng.com

ResourceManger Restart

- ResourceManager是管理资源和调度运行在YARN上的应用程序的中央机构，因此在一个YARN集群中ResourceManager可能是单点故障的，即只存在一个ResourceManager，这样在该节点出现故障时，就需要尽快重启ResourceManager，以尽可能地减少损失。本文将学习ResourceManager重启的特性，该特性使ResourceManager在重启时可以继续运行，并且在ResourceManager处于故障时对最终用户不可见。
- ResourceManager重启可以划分为两个阶段。第一阶段，增强的ResourceManager（RM）将应用程序的状态和其它认证信息保存到一个插入式的状态存储中。RM重启时将从状态存储中重新加载这些信息，然后重新开始之前正在运行的应用程序，用户不需要重新提交应用程序。第二阶段，重启时通过从NodeManagers读取容器的状态和从ApplicationMasters读取容器的请求，集中重构RM的运行状态。与第一阶段不同的是，在第二阶段中，之前正在运行的应用程序将不会在RM重启后被杀死，所以应用程序不会因为RM中断而丢失工作。

www.ibeifeng.com

ResourceManger Restart

- RM在客户端提交应用时，将应用程序的元数据（如ApplicationSubmissionContext）保存到插入式的状态存储中，RM还保存应用程序的最终状态，如完成状态（失败, 被杀死, 执行成功），以及应用完成时的诊断。除此之外，RM还将在安全的环境中保存认证信息如安全密钥，令牌等。RM 任何时候关闭后，只要要求的信息（比如应用程序的元数据和运行在安全环境中的认证信息等）在状态存储中可用，在RM重启时，就可以从状态存储中获取应用程序的元数据然后重新提交应用。如果在RM关闭之前应用程序已经完成，不论是失败、被杀死还是执行成功，在RM重启后都不会再重新提交。

www.iberfeng.com

ResourceManger Restart

- NodeManagers和客户端在RM关闭期间将保持对RM的轮询，直到RM启动。当启动后，RM将通过心跳机制向正在与其会话的NodeManager和ApplicationMasters发送同步指令。目前NodeManager和ApplicationMaster处理该指令的方式为：NodeManager将杀死它管理的所有容器然后向RM重新注册，对于RM来说，这些重新注册的NodeManager与新加入的NodeManager相似。ApplicationMasters在接收到RM的同步指令后，将会关闭。在RM重启后，从状态存储中加载应用元数据和认证信息并放入内存后，RM将为每个还未完成的应用创建新的尝试。正如之前描述的，此种方式下之前正在运行的应用程序的工作将会丢失，因为它们已经被RM在重启后使用同步指令杀死了。

www.ibeifeng.com

本课程版权归北风网所有

欢迎访问我们的官方网站

www.ibeifeng.com

www.ibeifeng.com