法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：小象学院
  - 新浪微博：小象AI学院

大数据应用与实践（第一期）

大纲

- 大数据技术体系
- 如何学习大数据及就业方向
- Zookeeepr设计原理、应用场景
- 实战：Zookeeper搭建和常用管理操作

大纲

大数据技术体系

大数据技术体系


数据可视化（Cboard、Echart）
数据展示
数据仓库（Hive）
数据挖掘（SparkMLLib）
OLAP（Druid）
数据分析
批处理（MapReduce）
流处理（SparkStreaming）
交互式分析（SparkSQL）
计算框架
资源管理（YARN、Mesos）
资源管理
数据存储（HDFS、Kafka、Redis、Hbase、MySQL）
数据存储
日志收集（Flume、Logstash、Scribe）
数据收集
数据源（互联网、智能设备、社交、电商等）

大数据应用领域

- 互联网领域

  搜索引擎，推荐系统，广告系统

- 电商领域

  用户画像，推荐系统，用户行为分析

- 医疗领域

  流行病预测，病情分析

- 视频领域

  用户标签系统，视频分析，广告系统

- 金融领域

  风控系统，欺诈分析

大纲

如何学习大数据及就业方向

如何学习大数据

1. 良好的自主学习能力和动手能力
2. 系统的了解大数据生态系统技术框架
3. 找到学习的切入点，不断拓展知识结构的广度
   - 大数据平台开发
   - 数据分析
4. 抓住一个技术方向，不断的深入研究，增加知识结构的深度
5. 主动学习探索新知识
6. 定期知识梳理

大数据必备技能

- 编程语言：Java/Python/Scala
- Linux常用命令、Shell编程
- HDFS原理、MapReduce原理及编程、YARN原理、Hadoop集群搭建
- Hive原理、HQL、自定义函数、数据仓库的设计
- Spark原理、SparkStreaming编程、SparkSQL
- Kafka原理、配置搭建、JavaAPI
- Flume原理、搭建
- Zookeeper原理、搭建

大数据就业方向

- 大数据分析工程师
  - 数据仓库设计与实现
  - 理解业务完成数据分析需求，开发报表
- 大数据平台开发工程师
  - 日志分析系统
  - 推荐系统
  - 用户标签系统
- 大数据运维工程师
  - 各种大数据框架集群搭建与集群运维
  - 技术框架版本升级，问题排查处理

大数据就业方向

- 大数据算法工程师
  - 使用大数据技术进行算法模型训练
  - 利用算法解决复杂业务问题
- 大数据内核开发工程师
  - 优化大数据技术框架内核
  - 开发公司内部大数据系统
  - 在公司内部推广新的技术方案

大数据就业方向

美团点评零售事业部招聘

Hadoop开发工程师

25k-50k /北京 / 经验3-5年 / 本科及以上 / 全职

hadoop 数据开发 大数据 数据挖掘

10:40 发布于拉勾网

职位要求：

1）计算机相关专

2）熟悉 Linux下

3）精通hadoop

4）较为丰富的数
能力。

5）思维敏捷，有

工作地址

北京 - 海淀区 - 中

职位诱惑：

免费班车,技术氛围,发展平台

职位描述：

工作职责
1、参与打造数据中内容的规划、设计、开发和优化工作，实现高质量数据的互通与共享；
2、参与数据模型体系构建及数据主题设计和开发，搭建离线、实时数据公共层；
3、参与数据产品与应用的数据研发，发掘数据商业价值，打造极致体验的数据产品；
4、深入理解数据产品的使用场景，为业务方在可用性、成本上做更好的设计做参考。

任职要求
1、熟悉数据仓库建模理论，3年以上相关领域实践经验；
2、熟悉Hadoop、Hive、Spark等大数据技术；
3、熟练使用Python/Java或其他语言进行复杂业务逻辑的数据处理工作，具备海量数据处理以及性能优化的能力；
4、对MySQL、Redis、HBase等数据库有一定的了解和使用经验；
5、对olap，多维分析及kylin熟悉的更好；
6、思路清晰，具备良好的沟通能力和理解能力，较强的学习能力以及快速解决问题的能力；
7、对新技术，新事物有很好的探索和求知欲。

目录

Zookeeepr设计原理

分布式系统

- 什么是分布式系统

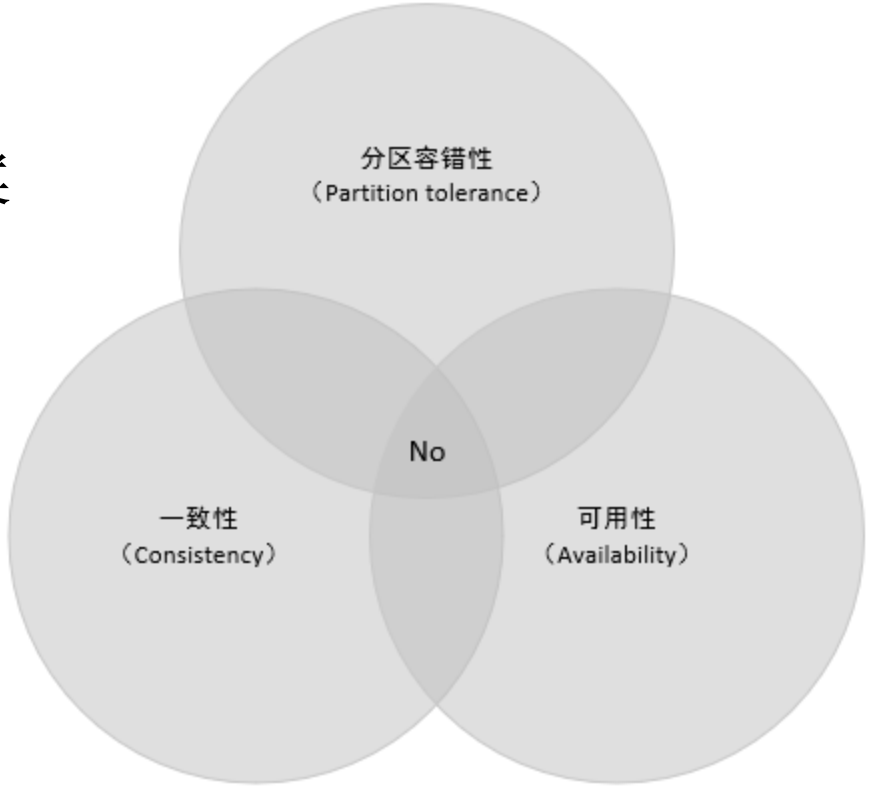
  《分布式系统概念与设计》一书定义分布式系统是一个**硬件**或**软件**组件分布在**不同**的**网络计算机**上，彼此之间通过**消息传递**进行**通信**和**协调**的系统

分布式系统

- 分布式系统的特点
  - 分布性
  - 对等性
  - 并发性
  - 缺乏全局时钟
  - 故障总是会发生
- 分布式环境的各种问题
  - 通信异常
  - 网络分区（俗称“脑裂”）
  - 三态（成功、失败、超时）
  - 节点故障

分布式系统-CAP定理

- 分区容错性（Partition tolerance ）
- 一致性(Consistency)
- 可用性(Available)
- 一致性和可用性平衡方案
  - 最终一致性
  - 案例：Zookeeper



Zookeeper简介

- 一个开源的针对大型分布式系统的**可靠**协调系统
- 设计目标是：将复杂且容易出错的分布式式一致性服务封装起来，构成一个高效可靠的原语集，并以简单易用的接口提供给用户使用。
- 提供的功能包括：发布/订阅，分布式协调/通知，配置管理，集群管理，主从协调，分布式锁等。

为什么选择Zookeeper

- 开源、免费
- 高效、可靠的解决数据一致性问题
- 工业界大型分布式系统广泛应用
- 简单易用

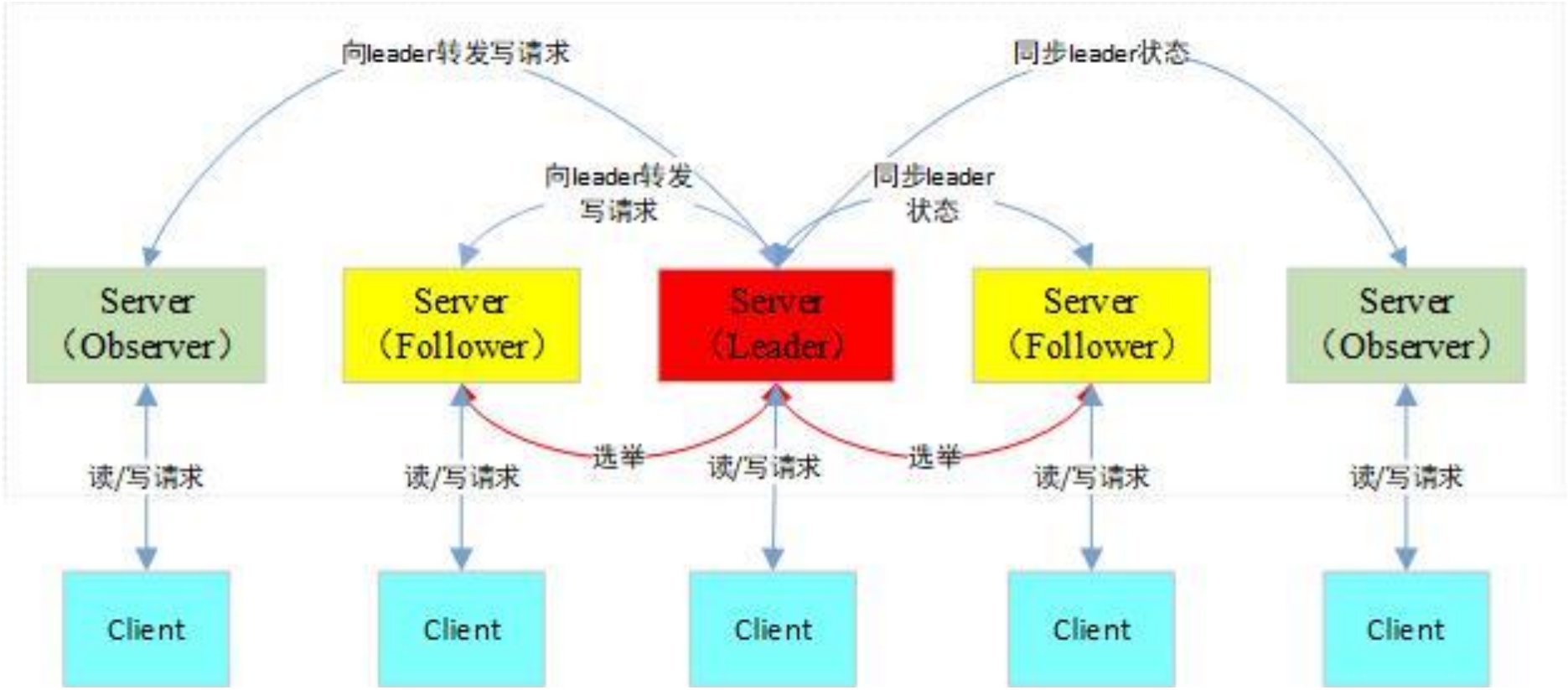
Zookeeper特性

- 最终一致性
  - 保证最终数据能够达到一致，这是Zookeeper最重要的功能。
- 顺序性
  - 从同一个客户端发起的事务请求，最终会严格地按照其发送顺序被应用到Zookeeper中。
- 可靠性
  - 一旦服务器成功的应用一个事务，并完成了客户端的响应，那么该事务所引起的服务端状态变更将会被一直保留下去。

Zookeeper特性

- 实时性
  - Zookeeper不能保证两个客户端能同时得到刚更新的数据，如果需要最新数据，应该在读数据之前调用sync()接口。
- 原子性：
  - 一次数据更新要么成功，要么失败。
- 单一视图：
  - 无论客户端连接到哪个服务器，看到的数据模型都是一致的。

Zookeeper架构

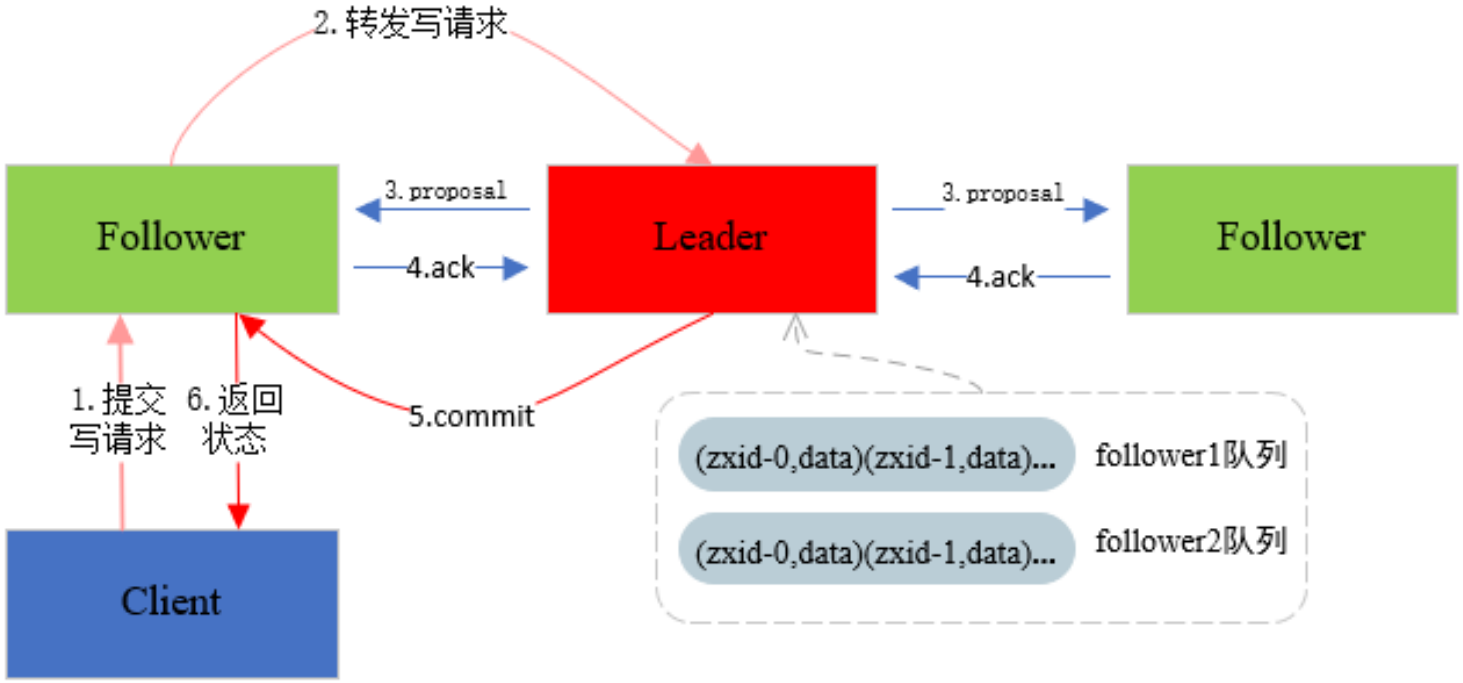


zookeeper角色

Leader		更新系统状态，处理事务请求，负责进行投票的发起和决议
Leaner	Follower	处理客户端非事务请求并向客户端返回结果，将写事务请求转发给Leader，同步Leader的状态，选主过程中参与投票。
	Observer	接收客户端读请求，将客户端写请求转发给Leader，不参与投票过程，只同步Leader的状态。目的是为了扩展系统，提高读取速度。
Client		请求发起方。

Zookeeper写入

1. 数据写入最终一致性核心算法ZAB算法
2. Leader负责处理写事务请求
3. Follower负责向Leader转发写请求，响应Leader发出的提议

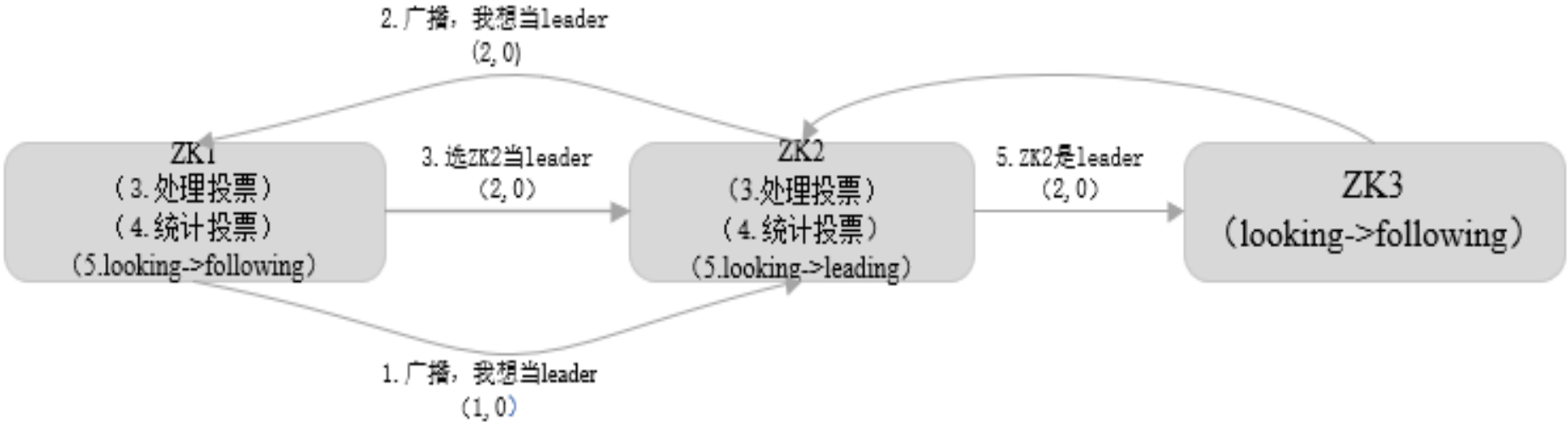


Zookeeper选举

- 服务器四种状态：
  - LOOKING：寻找Leader状态，处于该状态需要进入选举流程
  - LEADING：领导者状态，表明当前服务角色为Leader
  - FOLLOWING：跟随者状态，Leader已经选举出来，表明当前服务角色为Follower
  - OBSERVER：观察者状态，表明当前服务角色Observer
- 事务ID：用ZXID表示，是一个64位的数字，由Leader统一分配，全局唯一，不断递增。

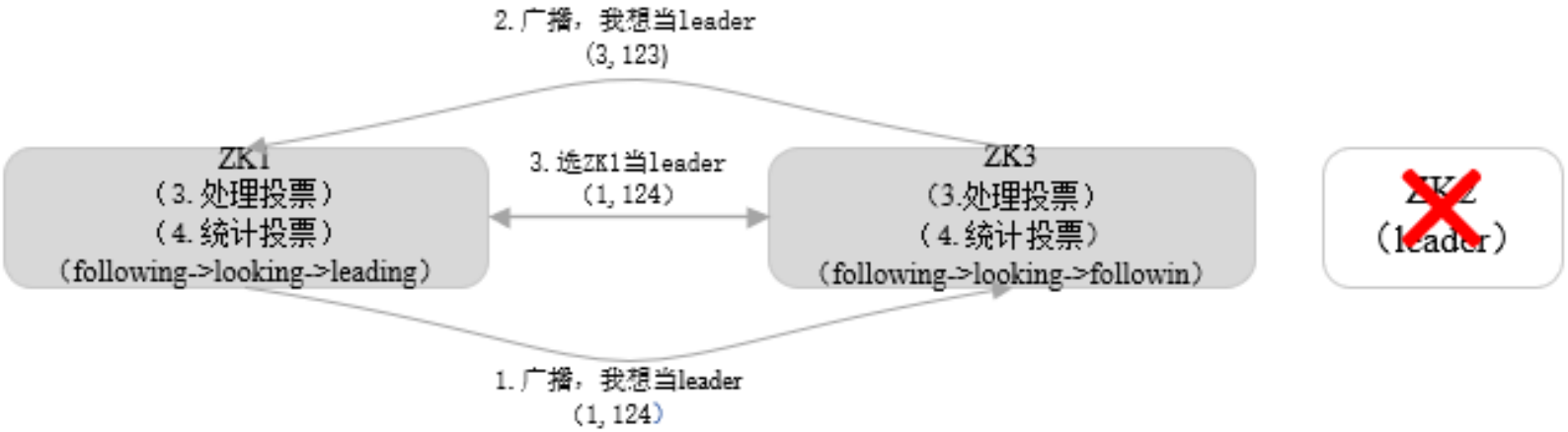
Zookeeper选举（全新启动期间）

1. 每个Server发出一个投票,内容为（myid，ZXID）
2. 接收来自各个Server的投票
3. 处理投票
4. 统计投票
5. 改变服务器状态



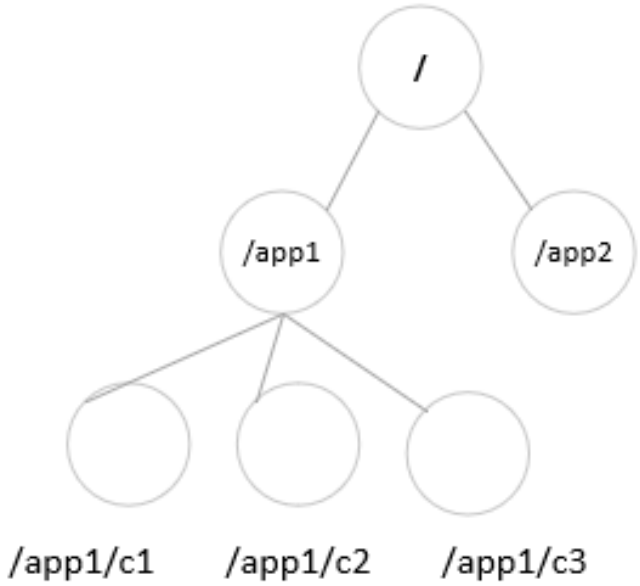
Zookeeper选举（运行期间）

1. 所有Server切换状态为LOOKING，每个Server发出一个投票，内容为（myid，ZXID）
2. 接收来自各个Server的投票
3. 处理投票
4. 统计投票
5. 改变服务器状态



数据模型Znode

- Zookeeper特有的数据节点Znode，视图结构类似Linux文件系统，没有目录和文件的概念
- Znode是Zookeeper中数据的最小单元
- Znode上可以保存数据，通过挂载子节点构成一个树状的层次化命名空间
- Znode树的根由“/”斜杠开始



Znode - 节点类型

- 节点类型
  - 持久节点（PERSISTENT）
  - 临时节点（EPHEMERAL）
  - 顺序节点（SEQUENTIAL）
- 组合节点类型
  - 持久节点（PERSISTENT）
  - 持久顺序节点（ PERSISTENT _ SEQUENTIAL ）
  - 临时节点（EPHEMERAL）
  - 临时顺序节点（EPHEMERAL_ SEQUENTIAL）

Znode - 版本

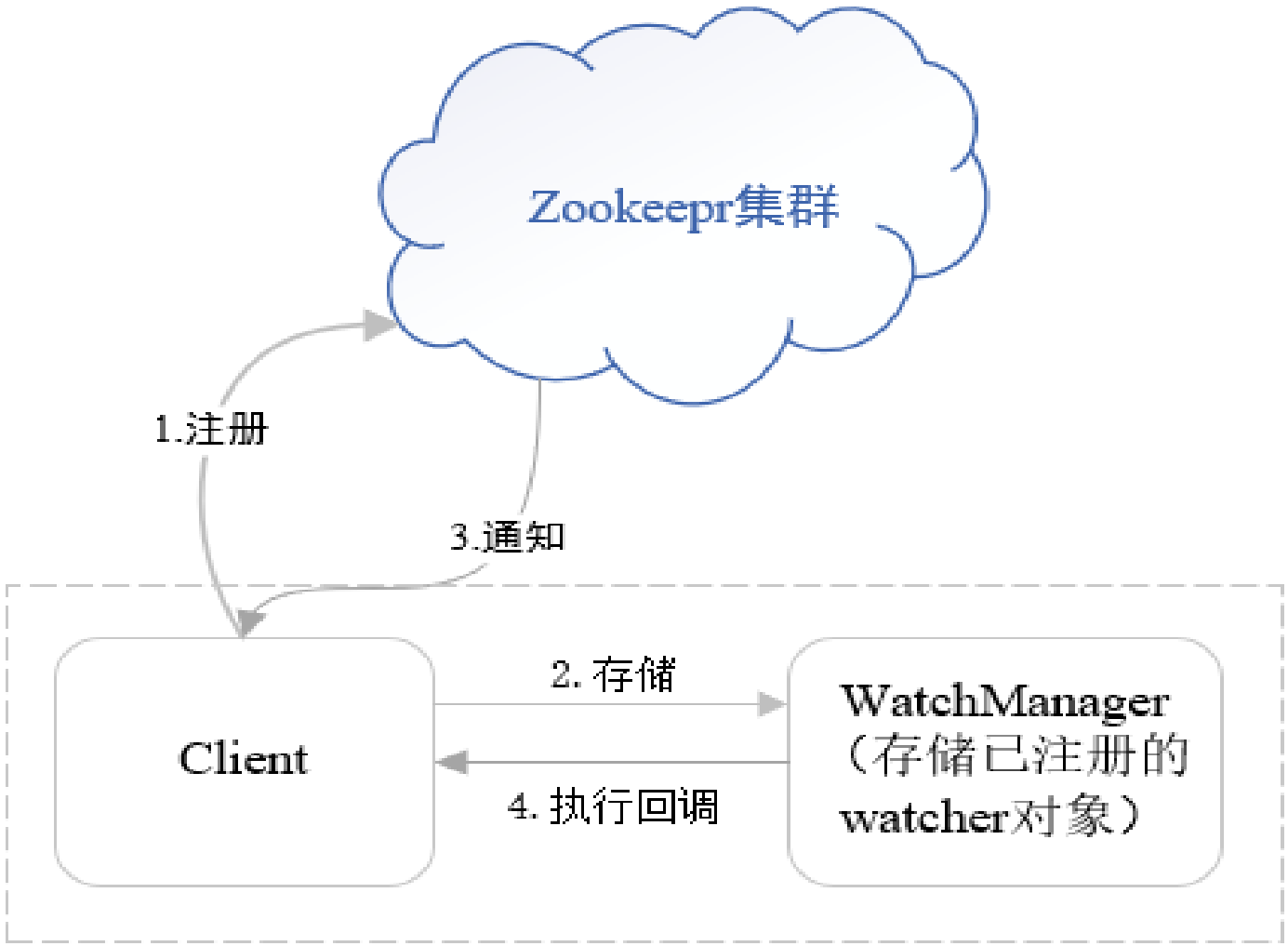
- 版本类型
  - dataVersion：当前数据节点数据内容的版本号
  - cVersion：当前数据节点子节点的版本号
  - aVersion：当前数据节点ACL权限变更版本号
- 如何保证分布式数据原子性操作
  - 悲观锁
  - 乐观锁
  - 使用version实现乐观锁机制中的“写入校验”

```
//获取当前请求中的version
version = setDataRequest.getVersion();
//获取当前服务器上该数据的最新version
int currentVersion = nodeRecord.stat.getVersion();
if(version != -1 && version != currentVersion){
    throw new KeeperException.BadVersionException(path);
}
version = currentVersion + 1;
```

Znode - 状态

状态属性	说明
cZxid	Znode被创建时的事务ID
ctime	Znode被创建的时间
mZxid	Znode最后一次被更新的事务ID
mtime	Znode最后一次被更新的时间
pZxid	Znode子节点列表最后一次被修改时的事务ID
cversion	子节点的版本号
dataVersion	数据节点的版本号
aclVersion	节点的ACL版本号
ephemeralOwner	创建该Znode的会话的sessionID，持久节点该值为0
dataLength	数据内容长度
numChildren	当前Znode的子节点个数

Znode - Watcher机制



目录

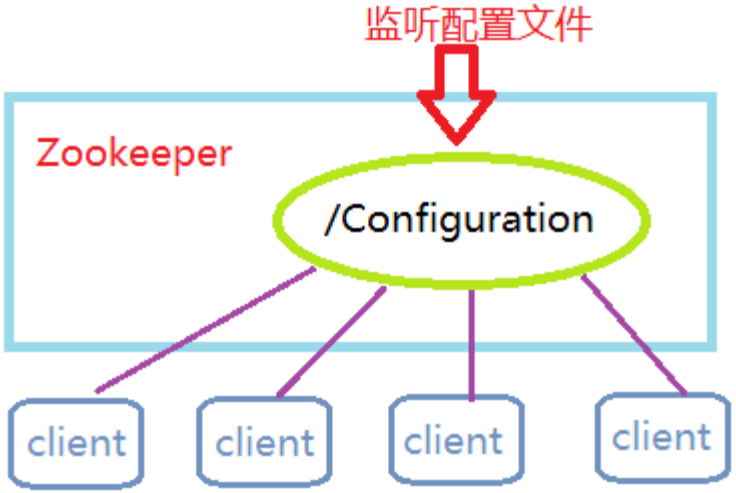
Zookeeepr应用场景

统一命名服务

- 分布式环境下，经常需要对应用/服务进行统一命名，便于识别不同服务；
  - 类似于域名与ip之间对应关系，域名容易记住；
  - 通过名称来获取资源或服务的地址，提供者等信息
- 按照层次结构组织服务/应用名称
  - 可将服务名称以及地址信息写到Zookeepr上，客户端通过Zookeeper获取可用服务列表
- 全局唯一ID
  - 依赖Zookeeper的顺序节点可以轻松生成全局唯一ID

配置管理

- 分布式环境下，配置文件管理和同步是一个常见问题
  - 集群中所有节点的配置信息是一致的，比如Hadoop
  - 对配置文件修改后，希望能够快速同步到各个节点上
- 配置管理可交由Zookeeper实现；
  - 可将配置信息写入Zookeeper的一个Znode上；
  - 各个节点监听这个Znode
  - 一旦znode中的数据被修改，将通知各个节点更新。



集群管理

- 分布式系统中，实时掌握每个节点的状态是有必要的
  - 可以根据节点实时状态做出一些调整
- 可交由Zookeeper实现
  - 将节点信息写入到一个Znode上
  - 监听这个Znode可获取它的实时状态变化
- 节点动态上下线
- Master选举

分布式锁

- Zookeeper是强一致的；
  - 多个客户端同时在Zookeeper上创建相同znode，只有一个创建成功。
- 实现锁的独占性
  - 多个客户端同时在Zookeeper上创建相同znode，创建成功的那个客户端得到锁，其他客户端等待。
- 控制锁的时序
  - 各个客户端在某个znode下创建临时znode（类型为CreateMode.EPHEMERAL_SEQUENTIAL），这样，该znode可掌握全局访问时序。

分布式队列

- 先入先出队列
  - 队列按照FIFO的方式进行入队和出队操作，例如实现生产者和消费者的模型（可用分布式锁实现）。
- 同步队列
  - 当一个队列的成员都聚齐时，这个队列才可用，否则一直等待所有成员到达，这种是同步队列。
  - 一个聚合计算开始之前需要并行计算的很多子任务全部完成才能进行
  - 为这个聚合计算创建一个Znode，同时监听这个Znode，其他并行子任务在这个Znode下创建一个临时节点，每次Znode变都会发出通知，接收到通知统计子节点个数，直到达到所有的子任务数。

大纲

实战：Zookeeper搭建与常用操作

Zookeeper安装步骤

- 下载压缩包

  http://mirrors.hust.edu.cn/apache/zookeeper

- 上传服务器，解压

  su – hadoop（切换到hadoop用户）

  tar -zxvf zookeeper-3.4.10.tar.gz（解压）

- 设置环境变量

  su – root(切换用户到root)

  创建软连接

  ln -s /bigdata/zookeeper-3.4.10 /usr/local/zookeeper

Zookeeper安装步骤

添加环境变量：

vi /etc/profile

export ZOOKEEPER_HOME=/usr/local/zookeeper

export PATH=$PATH:$ZOOKEEPER_HOME/bin

重新编译文件，使环境变量生效

source /etc/profile

Zookeeper安装步骤

- 修改配置文件

  切换到hadoop账号

  su - hadoop

  进入到zookeeper安装目录的conf目录

  cd /usr/local/zookeeper/conf

  cp zoo_sample.cfg zoo.cfg

Zookeeper安装步骤

在zoo.cfg配置文件中添加如下内容：

#快照文件存储目录

dataDir=/usr/local/zookeeper/data

#事务日志文件目录

dataLogDir=/usr/local/zookeeper/log

server.1=node01:2888:3888 (主机名:心跳端口:选举通信端口)

server.2=node02:2888:3888

server.3=node03:2888:3888

Zookeeper安装步骤

- 在zookeeper的安装目录下创建data和log文件夹

  mkdir -m 755 data

  mkdir -m 755 log

- 在data文件夹下新建myid文件，myid的文件内容为从1开始的数字，每个安装节点创建一个相同的文件，数字逐渐递增加1

  echo 1 > /usr/local/zookeeper/data/myid

Zookeeper安装步骤

- 将安装配置好的Zoopeeper拷贝到其他需要安装的节点

  scp -r /bigdata/zookeeper-3.4.10 hadoop@node2:/bigdata/

  scp -r /bigdata/zookeeper-3.4.10 hadoop@node3:/bigdata/

- 修改node2和node3的myid文件

  node2节点

  echo 2 > /usr/local/zookeeper/data/myid

  node3节点

  echo 3 > /usr/local/zookeeper/data/myid

Zookeeper安装步骤

- 启动Zookeeper集群

  分别启动三个节点的Zookeeper

  /usr/local/zookeeper/bin/zkServer.sh start

  查看集群状态

  zkServer.sh status

  关闭Zookeeper

  /usr/local/zookeeper/bin/zkServer.sh stop

集群启动/停止脚本

启动脚本：zkStart-all.sh
内容如下：

```
#!/bin/bash
echo "start zkserver..."
for i in 1 2 3
do
ssh node0$i "source /etc/profile;/usr/local/zookeeper/bin/zkServer.sh start"
done
echo "zkServer started!"
```

停止脚本：zkStop-all.sh
内容如下：

```
#!/bin/bash
echo "stop zkserver..."
for i in 1 2 3
do
ssh node0$i "source /etc/profile;/usr/local/zookeeper/bin/zkServer.sh stop"
done
echo "zkServer stoped!"
```

大纲

运维管理

Zookeeper命令行操作

- zkCli.sh -server hostname:port

  进入命令行客户端工具，如果不指定server地址，则直接连接本地的zookeeper

- help查看帮助

- ls path [watch]

  查看path路径的znode信息，添加watch选项可以监听znode树形结构的变化

- create [-s] [-e] path data acl

  创建一个新的 znode ，-s表示创建顺序节点，-e表示创建临时节点， path表示创建的znode所在路径，data表示创建的znode的内容数据，acl表示权限控制

Zookeeper命令行操作

- get path [watch]

  获取Znode信息，添加watch选项可以监听znode数据内容的变化

- set path data [version]

  更新Znode数据

- delete path [version]

  删除无子节点的Znode节点

- rmr path

  删除节点，有子节点的Znode 也可以删除

Zookeeper四字命令

- conf

  输出Zookeeper运行时使用的基本配置信息

- cons

  输出当前服务器上所有客户端连接的详细信息

- stat

  输出Zookeeper运行时状态信息

- srvr

  与stat命令功能一致，区别是不会将客户端的连接情况输出，仅输出服务器的自身信息

Zookeeper四字命令

- wchs

  输出当前服务器上管理的Watcher的概要信息

- wchc

  输出当前服务器上管理的Watcher的详细信息

- mntr

  输出比stat命令更详尽的服务器统计信息

- envi

  输出Zookeeper所在服务器运行时的环境信息

疑问

- 问题答疑：http://wenda.chinahadoop.cn
  - 可邀请老师或者其他人回答问题

作业

1. 总结课上笔记，将个人总结完的笔记提交到作业的小程序，目前支持简书和知乎。
2. 搭建Zookeeper集群，课上Zookeeper的练习自己操作一遍，将操作过程截图上传到作业。

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：小象学院
- 新浪微博：小象AI学院