ZooKeeper浅析

**背景**

随着互联网技术的高速发展，企业对计算机系统的计算和存储能力要求越来越高，并出现了高并发、海量存储等词汇。在这样的背景下，单独依靠少量高性能主机来完成计算任务已不能满足企业的需求，企业的IT架构已逐步从集中式向分布式过渡。所谓的分布式就是把一个计算任务分解成若干个计算单元，并分派到不同的计算机中去执行，然后汇总计算结果的过程。

**ZooKeeper是什么？**

ZooKeeper是源码开放的分布式协调服务，由雅虎创建，是Google Chubby的开源实现。ZooKeeper是一个高性能的分布式数据一致性解决方案，它将复杂的、容易出错的分布式一致性服务封装起来，构成一个高可靠的原语集，并提供一系列简单易用的接口给用户使用。其主要特点如下：

* 主要特点
  + 源码开放
  + 数据一致性
  + 实时性
  + 高性能
  + 应用广泛：Hadoop、Hbase、Storm等。

**ZooKeeper的基本概念**

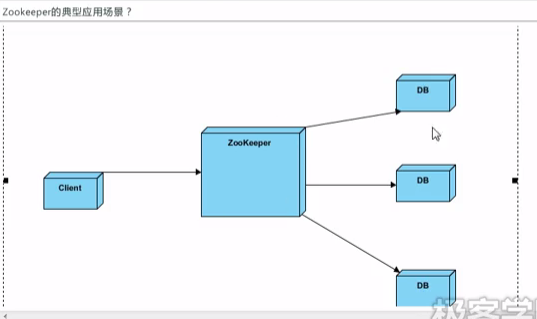
1. 集群角色
   1. Leader：整个集群工作的核心
   2. Follower：ZooKeeper集群状态的跟随者
   3. Observer：一个观察者角色
2. 会话：指客户端和ZooKeeper服务器建立的长连接。通过这个连接，客户端能够通过心跳检测与服务器保持有效会话。
3. 数据节点
   1. 集群中的一台机器称为一个节点
   2. 分为永久节点和临时节点，可以保存信息
4. 版本
   1. version：当前数据节点内容的版本号
   2. cversion：当前数据节点子节点的版本号
   3. sversion：当前数据节点ACL变更版本号
5. 事件监听器：ZooKeeper允许用户在指定节点上注册一些watcher，当数据节点发生变化时，ZooKeeper服务器会把变化的通知发送给感兴趣的客户端。
6. ACL权限控制：ZooKeeper采用ACL策略来进行权限控制。

**ZooKeeper的典型应用场景**

**1. 数据发布/订阅**

发布/订阅顾名思义就是一方把数据发布出来，另一方通过某种手段可以得到数据。通常数据订阅有两种方式：推模式和拉模式。ZooKeeper采用将推拉两种方式相结合的手段，即发布者将数据发布到ZooKeeper集群的节点上，订阅者通过一定的方法告诉服务器其对哪个节点的数据感兴趣，当这些节点的数据发生变化时，服务器就会通知客户端，客户端得到通知后可以去服务器获取数据信息。

**2. 负载均衡**



* 在DB启动的时候先到ZooKeeper中注册成临时节点，ZooKeeper的节点有两种：临时节点和永远节点。当临时节点在服务器出现问题时，节点会自动从ZooKeeper上删除，这样ZooKeeper上的服务器列表就是最新的并可用的。
* 客户端需要读写数据库时，它会从ZooKeeper得到所有可用的DB连接信息。
* 客户端随机选择一个并与之建立连接。
* 当客户端发现连接不可用时，会再随机得选择一个可用连接。

**3. 命名服务**

顾名思义，就是提供名称服务。例如数据库表格ID，一般是自动增长的或者是UUID，但二者各有缺陷：自动增长的ID局限在单表单库中使用，不能在分布式中使用，而UUID可以在分布式中使用但没有规律又难以理解。而我们可以借助ZooKeeper来生成一个顺序增长的，同时可以在分布式集群环境中使用的且命名易于理解的ID。

**4. 分布式协调/通知**

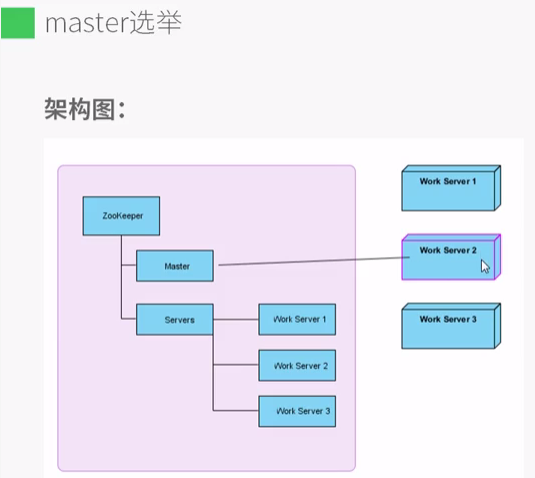
此处主要介绍该应用的心跳检测和master选举功能。

**心跳检测**

在分布式系统中，我们常常需要知道某个机器是否可用。在传统开发中，可以通过ping某个主机来实现，ping通说明对方可用，反之则不可用。而在ZooKeeper中，会让所有机器都注册成一个临时节点。我们判断一个机器是否可用，只需判断这个节点在ZooKeeper是否存在就可以了，不需直接去检查机器，降低系统复杂度。

**master选举**

假如有个系统A，它向外提供服务B，并且这个服务必须以7\*24小时不间断的向外提供服务，也就是说提供服务的机器不能有单点故障。于是我们考虑使用集群，我们采用的是master/slave的架构方式，集群中有一台主机和多台备机，由主机向外提供服务，备机负责监听主机的状态，一旦主机宕机，备机必须很迅速得接管主机并继续对外提供服务，在这个应用中，从备机选出一台主机的过程就是master选举。利用zookeeper可以很轻松得实现master选举。架构图如下：



左边紫色的区域代表zookeeper集群，右边3个立方体代表三台工作服务器，他们在各自的启动过程中，首先会去zookeeper的servers节点下创建自己的临时节点，并把自己的基本信息写入到这个临时节点。这个过程叫做服务注册。系统中的其它服务可以通过获取servers这个节点的子节点列表，来了解当前系统哪些服务器可用，这个过程叫做服务发现。紧接着，这些服务器会尝试着去创建master节点，谁能够创建成功谁就是master。假设这里2号服务器可以创建成功，那么2号服务器就会作为master向外提供服务。其它两台机器就作为slave，所有的workserver都必须关注master节点的删除事件，就是说，我们可以通过监听master节点的删除事件来了解master服务器是否宕机。一旦master宕机，我们会开始新一轮的选主工作，推选出一台新的master继续对外提供服务。

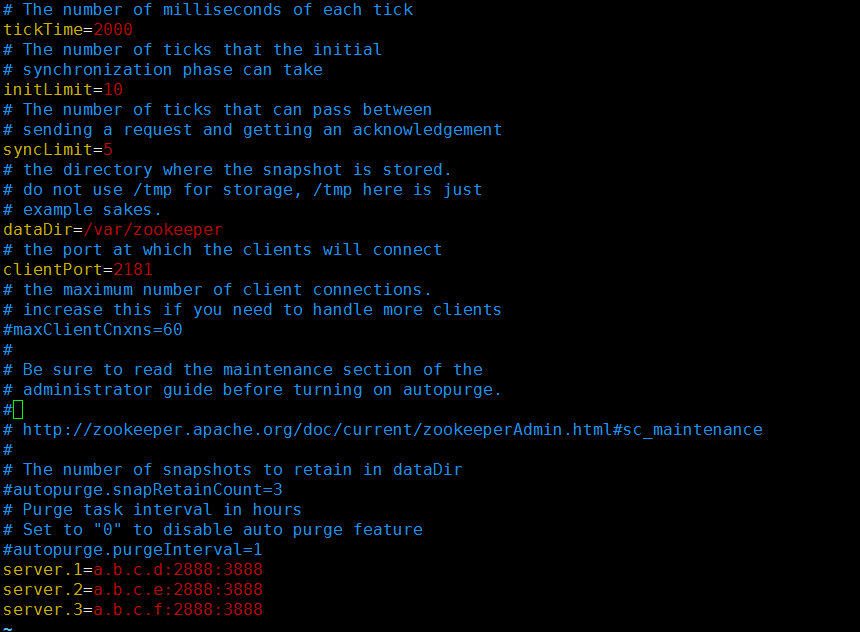
**ZooKeeper集群搭建**

**环境准备**

* VMWare虚拟机
* centos7
* SSH
* JDK
* python-setuptools
* zkpython

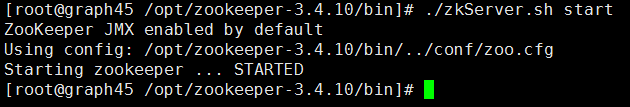
**步骤**

1. 使用VMWare创建3个centos7虚拟机，IP地址分别为a.b.c.d、a.b.c.e和a.b.c.f
2. 使用SSH连接虚拟机
3. 安装配置JDK、python-setuptools和zkpython(略)
4. 安装ZooKeeper
5. wget http://mirror.bit.edu.cn/apache/zookeeper/stable/zookeeper-3.4.10.tar.gz
6. tar zxvf zookeeper-3.4.10.tar.gz
7. cd zookeeper-3.4.10/src/c
8. ./configure && make && make install
9. 配置ZooKeeper
10. cd /opt/zookeeper-3.4.10/conf && cp zoo\_sample.cfg zoo.cfg
11. 编辑配置文件zoo.cfg，配置如下图：



注明：在配置文件中，只需重点关心ClientPort、dataDir和server.id=host:port:port等。其中ClientPort为ZooKeeper监听的端口，dataDir为存放特定文件的目录，而id是服务器的id，是个整数，host是服务器的地址，第一个port是leader服务器和follower服务器的通信端口，第二个port专门用于leader通信过程的选举通信。

1. 创建配置中的dataDir目录，并在该目录下创建一个myid文件，此文件需写入一行数据，就是写入配置中server后面的id，如1，当集群中有多台机器时，就需要每台机器都如此做，只是server的id不同而已。
2. 启动ZooKeeper：进入相应目录并执行程序，得到相应如下图：
3. /opt/zookeeper-3.4.10/bin && ./zkServer.sh start



1. 停止ZooKeeper：进入相应目录并执行程序
2. /opt/zookeeper-3.4.10/bin && ./zkServer.sh stop

本文链接：<https://addops.cn/post/zookeeper.html>

-- EOF --