ZooKeeper是Google的一个Chubby开源的实现，是Hadoop的分布式协调服务，包含一个简单的原语集，分布式应用程序可以基于ZooKeeper实现同步服务、配置维护和命名服务等，保证分布式数据一致性

Leslie Lamport在1990年提出一个基于消息传递的一致性算法Paxos，Google的Chubby和Apache的ZooKeeper都是基于Paxos的理论实现的，Paxos是目前为止唯一的分布式一致性算法

ZooKeeper提供通用的分布式锁服务，用于协调分布式应用

ZooKeeper特点：最终一致性、可靠性、实时性、独立性、原子性和顺序性

ZooKeeper与Keepalived都可以做高可用HA，Keepalived通过配置文件配置使用主动检测节点存活机制，访问Keepalived的请求不能负载均衡，不保存数据，监控节点不好管理，采用优先级监控（没有协同工作，功能单一），可扩展性差， ZooKeeper通过配置文件配置和API调用，使用被动检测节点存活机制（其他集群访问ZooKeeper，访问请求可以负载均衡），可以分布式保存数据（为了读取速度数据保存在内存，定期持久化到指定的dataDir目录中）

ZK Server：ZK Server Cluster的每一节点

ZK Server Leader：发出ZooKeeper数据改变的请求，更新系统状态

ZK Server Follower：向Leader发送请求处理Leader的返回响应结果，转发Client请求返回Client结果

ZK Server Observer：接收客户端请求转发给Leader，Leader宕机挂掉不参与选举为Leader，为了扩展系统，提高读取速度，只同步Leader的系统数据状态

ZNode Change（Create/Delete/SetData…）：ZooKeeper数据改变的请求

Zxid（ZooKeeper Transaction Id）：ZooKeeper事务编号即ZooKeeper数据改变的请求（proposal）的编号保证事务的顺序一致性，编号为64位数字，高32位是epoch用来表示Leader，低32位用于递增计数

ZNode：ZooKeeper保存的一致性的数据

ZooKeeper工作原理：每个Server在内存中存储一份数据，ZooKeeper启动时，从实例中选举一个Leader（Paxos协议），Leader负责处理数据更新等操作，当超过一半数量的Follower或Leader在内存中成功修改数据更新操作才算完成

ZooKeeper的事件处理确保整个Hadoop/HBase集群中同一时间有一个活跃的NameNode/HMaster节点以及存储配置信息等

配置ZooKeeper：

修改ZooKeeper安装目录/conf/zoo.cfg配置文件

tickTime=2000 ZooKeeper心跳请求时间间隔，单位毫秒

dataDir=文件路径名 ZooKeeper数据保存数据目录，默认/tmp目录下开启重启数据会被清除

clientPort=2181 ZooKeeper服务器端口接收客户端请求

initLimit=5 ZooKeeper的Leader与Follower之间初始化连接的最长心跳时间间隔个数

syncLimit=2 ZooKeeper的Leader与Follower之间发送请求消息或者同步数据最长心跳时间间隔个数

[server.数值=主机名:2888:3888][:observer] 配置指定数值为ID的ZooKeeper集群服务器[指定为Observer]，默认两个端口2888（ZooKeeper的Leader与Follower之间交换请求信息的端口），3888（ZooKeeper的Leader宕机挂掉时从Follower中选举新Leader的通信端口），因为ZooKeeper的数据变化是需要超过一半数量（为了容错和效率）的Server数据变化成功才算成功，ZooKeeper集群服务器数量为了容错（偶数与对应奇数服务器容错允许宕机的服务器个数相同）和防脑裂（一半数量服务器与另一半数量服务器通信出现问题时都无法选举出Leader都无法正常工作）一般是大于等于3的奇数，

在dataDir指定的ZooKeeper数据保存路径下创建myid文件保存主机的服务ID，每台ZooKeeper集群节点使用zkServer.sh start/status/stop/restart 启动/查看状态/停止/重启 ZooKeeper命令，在执行命令的当前路径下会生产ZooKeeper日志文件ZooKeeper.out

ZooKeeper的核心是原子广播，实现原子广播机制的协议叫ZAB（ZooKeeper Atomic Broadcast），ZAB协议有两种模式：恢复模式（选举）和广播模式（同步）

ZooKeeper每个Server工作过程有三个状态：looking（当前Server搜寻Leader）、leading（当前Server为选举的Leader）和following（当前Server与Leader同步）

Follower循环处理Leader的消息：PING（心跳消息）、PROPOSAL（请求）、COMMIT（服务器最新一次请求的消息）、UPTODATE（表明同步成功）、REVALIDATE（根据Leader的REVALIDATE结果，关闭待revalidate的session或允许接收消息）、SYNC（返回同步SYNC最新的消息给Client）

ZooKeeper安装目录/bin/zkCli.sh [IP地址] 使用当前机器客户端访问[指定IP地址]默认当前机器上的ZooKeeper服务器

ls / 查看ZooKeeper集群存储的数据

help 查看ZooKeeper帮助命令

create [-s] [-e] 路径名 数据 在ZooKeeper中创建保存指定数据的指定路径名的[序列，sequential有重名文件会增加序列号进行创建][短暂的，默认持久的]文件数据，路径名是ZooKeeper集群内部的路径名与Linux系统的路径名不同

get 路径名 获取指定路径名的数据，cZxid（创建事务的编号）、ctime（创建事务的时间）、mZxid（最新一次修改事务的编号）、mtime（最新一次修改事务的时间）、pZxid、cversion、dataVersion（数据版本号）、aclVersion、ephemeralOwner（拥有者，0x0所有人）、dataLength（数据长度，字符数量）、numChildren（当前节点下的子节点数）

ZooKeeper的数据模型具有层次化的目录结构，每个节点（ZNode）有唯一的路径标识（与Linux系统目录相似但ZNode节点目录可以保存数据），ZNode可以包含数据（数据可以保存多个版本）和子节点（ephemeral类型的ZNode不能有子节点），ZNode不支持部分读写，只支持一次性完整读写

ZNode有两种类型ephemeral（短暂的临时的，在客户端会话结束后自动删除，ZooKeeper使用ephemeral特性做高可用HA）和persistent（持久的），类型在创建时指定不能修改

Java远程方法调用RMI(Remote Method Invocation)，LocateRegistry.createRegistry(端口号)Java线程开启指定端口号，Naming.rebind(RMI协议的URL地址，类对象)绑定指定类对象到URL地址上，Naming.lookup(RMI协议的URL地址)获取指定URL地址上绑定的类对象

Watcher（观察者）是ZooKeeper的一个核心功能，可以监控ZooKeeper目录节点的数据变化以及子目录的变化，监控到变化服务器会通知设置在指定目录节点上的Watcher，每个Watcher只会通知一次变化后续需要再次设置Watcher获取下次变化通知，设置观察的操作有exists、getChildren和getData，触发观察的操作有create、delete和setData

Java线程类CountDownLatch的CountDownLatch.countDown()方法对创建对象时指定的数字进行减少，减少到0时会跳过CountDownLatch.await()等待方法

阿里巴巴公司开源的高性能服务框架Dubbo底层原理是基于ZooKeeper实现的

ZooKeeper的分布式锁与进程锁和线程锁作用原理基本一致，都是防止单个资源同时被多个客户端访问，分布式锁、进程锁和线程锁分别是基于Zookeeper目录节点的数据变化、操作系统信号量和同一时刻只允许一个线程执行线程程序段实现