



(12)发明专利申请

(10)申请公布号 CN 106293954 A

(43)申请公布日 2017.01.04

(21)申请号 201610643266.7

(22)申请日 2016.08.08

(71)申请人 浪潮(北京)电子信息产业有限公司
地址 100085 北京市海淀区上地信息路2号
2-1号C栋1层

(72)发明人 毕敬强

(74)专利代理机构 北京集佳知识产权代理有限公司 11227

代理人 罗满

(51) Int. Cl.

G06F 9/52(2006.01)

H04L 29/08(2006.01)

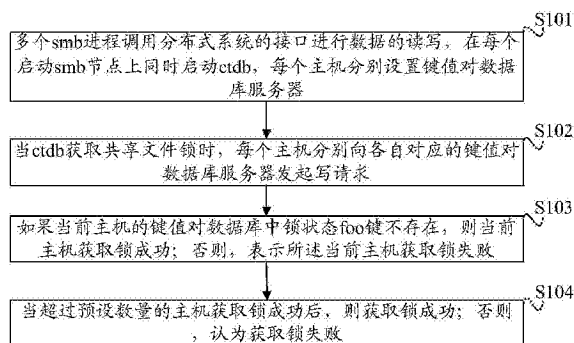
权利要求书1页 说明书4页 附图2页

(54)发明名称

一种基于分布式锁的高可用服务管理方法

(57)摘要

本发明公开了一种基于分布式锁的高可用服务管理方法,多个smb进程调用分布式系统的接口进行数据的读写,在每个启动smb节点上同时启动ctdb,每个主机分别设置键值对数据库服务器;当ctdb获取共享文件锁时,每个主机分别向各自对应的键值对数据库服务器发起写请求;如果当前主机的键值对数据库中锁状态foo键不存在,则当前主机获取锁成功;否则,表示当前主机获取锁失败;当超过预设数量的主机获取锁成功后,则获取锁成功;否则,认为获取锁失败。在本方案中ctdb不使用smb的后端分布式文件系统存放共享文件锁,所以分布式系统的故障等不会影响ctdb的正常使用;实现了smb的高可用和读写分布式文件系统的解耦合。



1. 一种基于分布式锁的高可用服务管理方法,其特征在于,包括:

多个smb进程调用分布式系统的接口进行数据的读写,在每个启动smb节点上同时启动ctdb,每个主机分别设置键值对数据库服务器;

当ctdb获取共享文件锁时,每个主机分别向各自对应的键值对数据库服务器发起写请求;

如果当前主机的键值对数据库中锁状态foo键不存在,则当前主机获取锁成功;否则,表示所述当前主机获取锁失败;

当超过预设数量的主机获取锁成功后,则获取锁成功;否则,认为获取锁失败。

2. 如权利要求1所述的基于分布式锁的高可用服务管理方法,其特征在于,在当前主机获取锁成功之后还包括:

将当前时刻写入所述键值对数据库的value中。

3. 如权利要求2所述的基于分布式锁的高可用服务管理方法,其特征在于,所述主机的数量为不少于3的奇数,所述预设数量大于所述主机的数量的一半。

4. 如权利要求3所述的基于分布式锁的高可用服务管理方法,其特征在于,所述认为获取锁失败包括:

重复获取的操作,直至失败次数达到预设次数后,认为获取锁失败。

5. 如权利要求4所述的基于分布式锁的高可用服务管理方法,其特征在于,所述预设次数为3次。

6. 如权利要求1至5任一项所述的基于分布式锁的高可用服务管理方法,其特征在于,在获取锁成功之后还包括:

当超过预设时间阈值后,锁状态过期,删除所述主机的键值对数据库服务器中key为foo的键值。

7. 如权利要求6所述的基于分布式锁的高可用服务管理方法,其特征在于,所述预设时间阈值为3秒。

一种基于分布式锁的高可用服务管理方法

技术领域

[0001] 本发明涉及高可用服务技术领域,特别是涉及一种基于分布式锁的高可用服务管理方法。

背景技术

[0002] 随着互联网时代的到来,微博、微信、网购等面向普通互联网用户的网站和应用正在蓬勃兴起,互联网界的巨头公司向数以亿计的用户提供着基于互联网的各种服务。遍布世界各地的互联网用户每天都在网上发布信息,他们产生的这些是个人计算机数据量的数倍。

[0003] 互联网公司通常采用高性能服务器来存储这些数据。但是普通的存储系统已经无法支撑越来越多的用户数据,而且断电、灾害和系统故障使得数据的安全性很难保证。为了随时应对激增的用户请求,越来越多的公司采用分布式存储系统,尤其是视频行业。分布式存储系统具有高可靠性、高可用性和高扩展性,可以避免由于单个节点失效而使整个系统崩溃的危险,可以将分布在各处的资源综合利用,同时可以将负载由单个节点转移到多个,从而提高了存储系统的性能。

[0004] smb软件调用分布式文件系统的接口并通过标准samba协议映射出远端的分布式文件系统。windows用户只需要挂载到本地,即可读写文件。在每个启动smb的节点上同时启动ctdb。多个ctdb组成一个master/slave类型的集群,从而可以实现smb进程高可用。

[0005] 每个启动smb进程的节点要启动ctdb进程。多个ctdb进程组成一个ctdb集群,用来实现smb的高可用。多个ctdb进程中有一个为master,其他为slave,master ctdb负责虚拟ip的分配、smb进程的启动等。多个ctdb进程为了实现协同,需要有一种方式实现角色的一致性。

[0006] 在传统方案中,ctdb的角色是通过获取分布式文件系统的锁文件来选举得到的。当有ctdb启动或者关闭时,所有的ctdb都会通过fcntl调用获取锁。如果分布式文件系统出现故障或者读写压力较大,ctdb获取锁会延迟较高甚至获取失败,这会严重影响smb进程的高可用性。

发明内容

[0007] 本发明的目的是提供一种基于分布式锁的高可用服务管理方法,目的在于解决现有技术中ctdb获取锁过程延迟较多,且容易因获取失败而严重影响smb进程的高可用性的问题。

[0008] 为解决上述技术问题,本发明提供一种基于分布式锁的高可用服务管理方法,包括:

[0009] 多个smb进程调用分布式系统的接口进行数据的读写,在每个启动smb节点上同时启动ctdb,每个主机分别设置键值对数据库服务器;

[0010] 当ctdb获取共享文件锁时,每个主机分别向各自对应的键值对数据库服务器发起

写请求；

[0011] 如果当前主机的键值对数据库中锁状态foo键不存在，则当前主机获取锁成功；否则，表示所述当前主机获取锁失败；

[0012] 当超过预设数量的主机获取锁成功后，则获取锁成功；否则，认为获取锁失败。

[0013] 可选地，在当前主机获取锁成功之后还包括：

[0014] 将当前时刻写入所述键值对数据库的value中。

[0015] 可选地，所述主机的数量为不少于3的奇数，所述预设数量大于所述主机的数量的一半。

[0016] 可选地，所述认为获取锁失败包括：

[0017] 重复获取的操作，直至失败次数达到预设次数后，认为获取锁失败。

[0018] 可选地，所述预设次数为3次。

[0019] 可选地，在获取锁成功之后还包括：

[0020] 当超过预设时间阈值后，锁状态过期，删除所述主机的键值对数据库服务器中key为foo的键值。

[0021] 可选地，所述预设时间阈值为3秒。

[0022] 本发明所提供的基于分布式锁的高可用服务管理方法，多个smb进程调用分布式系统的接口进行数据的读写，在每个启动smb节点上同时启动ctdb，每个主机分别设置键值对数据库服务器；当ctdb获取共享文件锁时，每个主机分别向各自对应的键值对数据库服务器发起写请求；如果当前主机的键值对数据库中锁状态foo键不存在，则当前主机获取锁成功；否则，表示当前主机获取锁失败；当超过预设数量的主机获取锁成功后，则获取锁成功；否则，认为获取锁失败。因为在本方案中ctdb不使用smb的后端分布式文件系统存放共享文件锁，所以分布式系统的故障等不会影响ctdb的正常使用；实现了smb的高可用和读写分布式文件系统的解耦合。本申请搭建简单，部署方便，适合切换不频繁的master/slave的分布式服务。

附图说明

[0023] 为了更清楚的说明本发明实施例或现有技术的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单的介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0024] 图1为本发明所提供的基于分布式锁的高可用服务管理方法的一种具体实施方式的流程图；

[0025] 图2为ctdb管理smb进程的示意图；

[0026] 图3为ctdb使用redis分布式锁的示意图。

具体实施方式

[0027] 为了使本技术领域的人员更好地理解本发明方案，下面结合附图和具体实施方式对本发明作进一步的详细说明。显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提

下所获得的所有其他实施例,都属于本发明保护的范围。

[0028] 本发明所提供的基于分布式锁的高可用服务管理方法的一种具体实施方式的流程图如图1所示,该方法包括:

[0029] 步骤S101:多个smb进程调用分布式系统的接口进行数据的读写,在每个启动smb节点上同时启动ctdb,每个主机分别设置键值对数据库服务器;

[0030] 步骤S102:当ctdb获取共享文件锁时,每个主机分别向各自对应的键值对数据库服务器发起写请求;

[0031] 步骤S103:如果当前主机的键值对数据库中锁状态foo键不存在,则当前主机获取锁成功;否则,表示所述当前主机获取锁失败;

[0032] 步骤S104:当超过预设数量的主机获取锁成功后,则获取锁成功;否则,认为获取锁失败。

[0033] 本发明所提供的基于分布式锁的高可用服务管理方法,多个smb进程调用分布式系统的接口进行数据的读写,在每个启动smb节点上同时启动ctdb,每个主机分别设置键值对数据库服务器;当ctdb获取共享文件锁时,每个主机分别向各自对应的键值对数据库服务器发起写请求;如果当前主机的键值对数据库中锁状态foo键不存在,则当前主机获取锁成功;否则,表示当前主机获取锁失败;当超过预设数量的主机获取锁成功后,则获取锁成功;否则,认为获取锁失败。因为在本方案中ctdb不使用smb的后端分布式文件系统存放共享文件锁,所以分布式系统的故障等不会影响ctdb的正常使用;实现了smb的高可用和读写分布式文件系统的解耦合。本申请搭建简单,部署方便,适合切换不频繁的master/slave的分布式服务。

[0034] 在上述实施例的基础上,本发明所提供的基于分布式锁的高可用服务管理方法中,在当前主机获取锁成功之后还包括:

[0035] 将当前时刻写入所述键值对数据库的value中。

[0036] 进一步地,主机的数量为不少于3的奇数,所述预设数量大于所述主机的数量的一半。主机的数量可根据实际需求进行设置,这均不影响本发明的实现。

[0037] 在此次认为获取锁失败后,重复获取的操作,直至失败次数达到预设次数后,认为获取锁失败。

[0038] 所述预设次数可以具体为3次。当然其他次数也可,并不限于3次。

[0039] 在上述任一实施例的基础上,为防止出现死锁,本发明实施例还可以进一步包括:

[0040] 当超过预设时间阈值后,锁状态过期,删除所述主机的键值对数据库服务器中key为foo的键值。

[0041] 其中,预设时间阈值可以为3秒。

[0042] 本发明实施例中主机数量以3个为例,对其具体工作过程进行进一步详细阐述。

[0043] 如图2 ctdb管理smb进程的示意图所示,ctdb使用的共享文件锁保存在分布式文件系统中。如图3 ctdb使用redis分布式锁的示意图所示,在3台主机上同时启动redis-server,使用redis kv数据库来保存锁状态。锁状态的保存形式是,key为foo,value为当前的时刻,精确到毫秒。

[0044] 如果ctdb尝试获取共享文件锁,则向这3台主机的redis-server依次发起写请求,如果foo这个键不存在,则写入value为当前的时刻,则表示获取锁成功,否则如果键foo存

在,则在当前主机获取锁失败。如果在2个以上主机获取锁成功,则获取锁成功,否则重试,失败次数达到3次,则认为获取锁失败。

[0045] 为防止出现死锁,每次获取锁结束3秒后,锁状态过期。3台主机的redis-server的key为foo的键值对都被删除。

[0046] 本申请所提供的基于分布式锁的高可用服务管理方法,实现了基于redis的分布式锁方案,具有下述优点:

[0047] (1)搭建简单,部署方便,非常适合切换不频繁的master/slave的分布式服务。

[0048] (2)因为在本方案中ctdb不使用smb的后端分布式文件系统存放共享文件锁,所以分布式系统的故障等不会影响ctdb的正常使用。

[0049] (3)实现了smb的高可用和读写分布式文件系统的解耦合

[0050] 本说明书中各个实施例采用递进的方式描述,每个实施例重点说明的都是与其它实施例的不同之处,各个实施例之间相同或相似部分互相参见即可。对于实施例公开的装置而言,由于其与实施例公开的方法相对应,所以描述的比较简单,相关之处参见方法部分说明即可。

[0051] 专业人员还可以进一步意识到,结合本文中所公开的实施例描述的各示例的单元及算法步骤,能够以电子硬件、计算机软件或者二者的结合来实现,为了清楚地说明硬件和软件的可互换性,在上述说明中已经按照功能一般性地描述了各示例的组成及步骤。这些功能究竟以硬件还是软件方式来执行,取决于技术方案的特定应用和设计约束条件。专业技术人员可以对每个特定的应用来使用不同方法来实现所描述的功能,但是这种实现不应认为超出本发明的范围。

[0052] 结合本文中所公开的实施例描述的方法或算法的步骤可以直接用硬件、处理器执行的软件模块,或者二者的结合来实施。软件模块可以置于随机存储器(RAM)、内存、只读存储器(ROM)、电可编程ROM、电可擦除可编程ROM、寄存器、硬盘、可移动磁盘、CD-ROM、或技术领域内所公知的任意其它形式的存储介质中。

[0053] 以上对本发明所提供的基于分布式锁的高可用服务管理方法进行了详细介绍。本文中应用了具体个例对本发明的原理及实施方式进行了阐述,以上实施例的说明只是用于帮助理解本发明的方法及其核心思想。应当指出,对于本技术领域的普通技术人员来说,在不脱离本发明原理的前提下,还可以对本发明进行若干改进和修饰,这些改进和修饰也落入本发明权利要求的保护范围内。

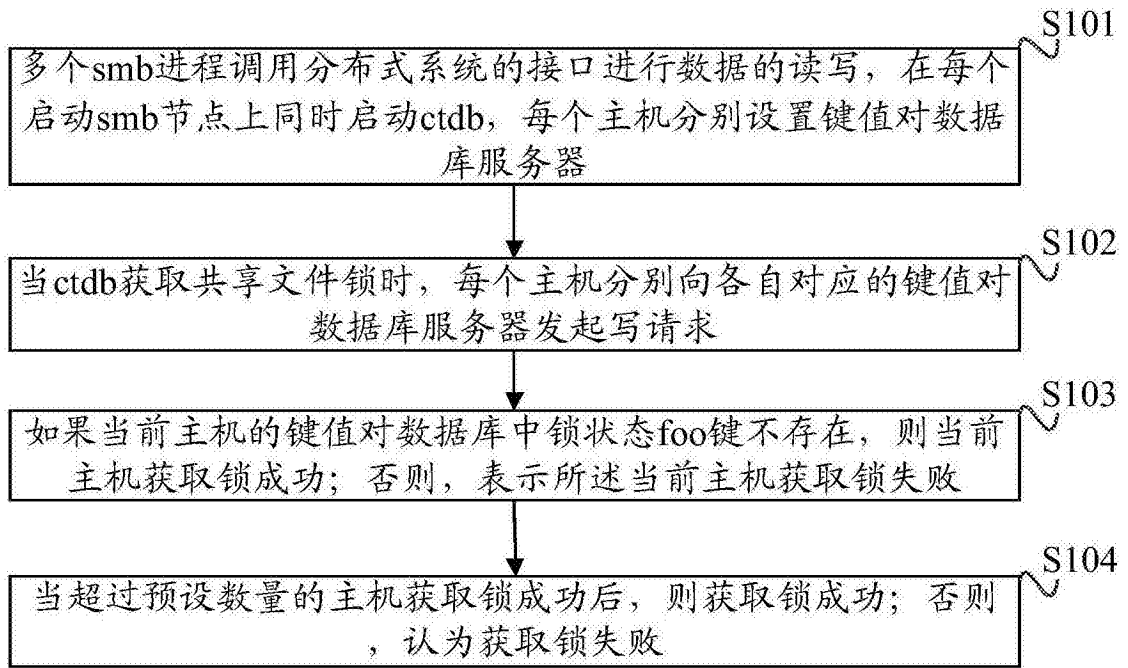


图1

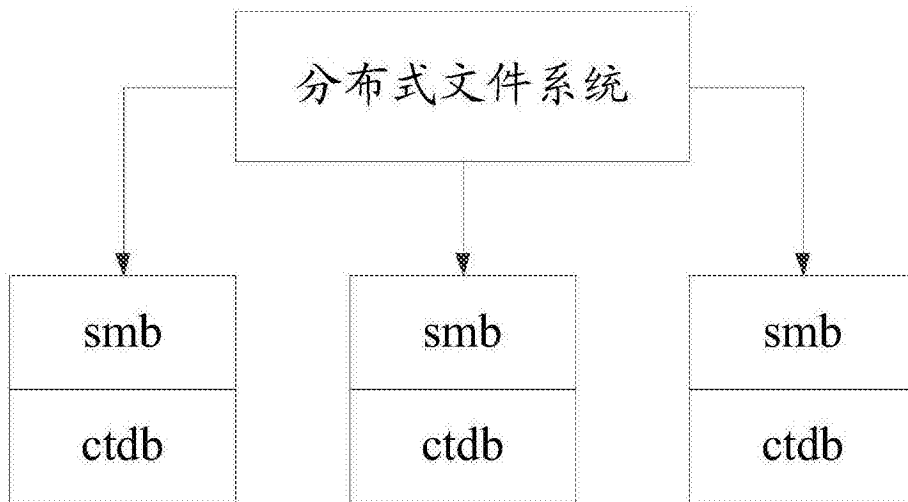


图2

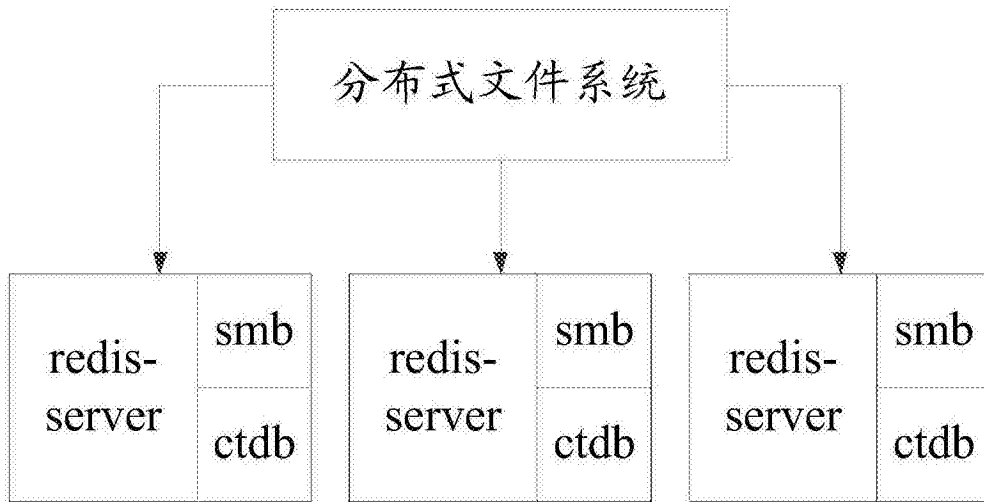


图3