可伸缩的元数据集群系统设计

刘　仲①　王晓东　周兴铭

(国防科学技术大学计算机学院　长沙 410073)

摘　要　目录子树分割方法与路径名散列方法是目前常见的两种元数据管理方法,它们都存在不均衡工作负载、大量元数据迁移、可伸缩性差等缺点。本文提出一种将元数据划分为目录路径元数据与文件自身元数据分开管理的新的元数据集群系统结构,并设计一种可伸缩的元数据管理模型。

关键词　分布式存储,元数据集群,目录路径,可伸缩

0　引　言

在传统的文件系统中,元数据与数据本身由同一个文件系统管理,保存在同一台存储设备上,并且为提高访问效率,元数据与其描述的数据在物理上尽可能的靠近。而在大规模分布式存储系统中,数据都是通过高速网络直接访问,元数据与用户数据分离,由独立的元数据服务器承担元数据的访问服务,尽管元数据的数据量相对于整个存储系统的数据容量而言比较小,但是有统计表明,在所有文件系统的访问中,对元数据的访问大约占全部访问次数的50%到80%[1],所以,高效的元数据管理对整个存储系统提供高性能和高可伸缩性至关重要。

在现有的分布式文件系统中,分配元数据的方法主要有两种,即目录子树分割(Directory subtree partitioning)方法[2]和纯散列(Pure hashing)方法[3]。目录子树分割根据目录子树来组织元数据,每个元数据服务器管理整个目录树中的一个或多个子目录树,如NFS、AFS、Coda、Sprite、LOCUS等都是用这种方法来管理元数据,其主要优点是,同一目录子树中的元数据保存在同一元数据服务器中,所以对该目录子树下文件的元数据获取只需要访问同一元数据服务器即可,能够有效利用用户端的缓存机制。其主要缺点是不能有效的平衡元数据服务器之间的工作负载,负载重的服务器可能成为系统瓶颈;在可伸缩性方面,若增加或减少元数据服务器,不能有效的重新平衡元数据服务器之间的工作负载;必须遍历文件路径的所有目录才能确定对该文件的访问权限,访问性能低效。纯散列技术通过散列算法分配元数据到不同服务器,解决了目录子树分割方法不能有效平衡元数据服务器之间工作负载的缺点。典型的散列计算方法是对文件全路径名进行散列计算来分配和定位元数据,优点是不需要遍历文件路径的所有目录就能直接定位文件的元数据;缺点是修改目录时,因为散列输入的改变,该目录下所有文件的元数据服务器必须更改,导致大量的元数据迁移;在可伸缩性方面,若增加或减少元数据服务器,则需要重新计算散列值从而导致大量的元数据迁移。Brandt[4]结合目录子树分割与散列方法的优点,基于文件路径名进行散列计算来快速定位文件的元数据,用一个访问控制链表来描述文件的访问权限,克服了必须遍历文件路径的所有目录才能确定该文件的访问权限的缺点。但是,依然存在当目录属性更改(目录名改变、删除,目录权限修改)或增加、删除元数据服务器时,需要重新计算散列值从而导致大量的元数据迁移的缺点。Lustre[5]有类似的缺点。

文件的目录路径用于实现文件系统的目录层次管理,文件的元数据用于记录文件的访问属性与数据对象的存储位置,将它们混在一起管理是现有元数据管理方法缺点的根源所在。受文件系统的inode设计思想启发,本文提出将文件的目录路径与元数据分开管理,目录路径包括文件所在的全目录路径名及相应的路径访问控制属性,目录路径通过类似于文件inode的目录路径inode来管理,目录路径inode的系统设计能够消除目录路径的修改(包括目录名修改与目录权限修改)对元数据的影响,根据文件名与其目录路径inode的动态线性散列算法,既具有纯散列算法的优点,又可以有效避免或大大减少

① 男,1971年生,博士生,副研究员;研究方向:网络存储,并行与分布处理;联系人。
(收稿日期:2004-04-09)

了元数据的更新与迁移。

1 元数据集群系统结构设计

1.1 OCFS 系统结构

我们提出的大规模集群文件系统(OSD-based Cluster File System,OCFS)的系统结构如图1,OCFS 包括4个部分,即客户端文件系统(Client File System,CFS)、目录路径索引服务器(Directory Path Index Server,DPIS)、元数据服务器(Medadata Server,MDS)和对象存储体(Object Storage Target,OST),各个部分之间通过高速网络互连。

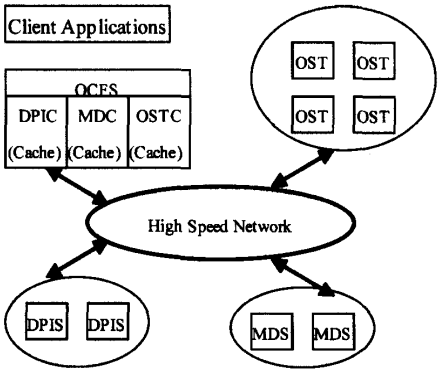

图1 OCFS 系统结构

在 OCFS 中,任何文件的数据分为三部分:目录路径元数据、文件本身元数据与数据对象。目录路径元数据包括文件所在的目录路径名和路径访问控制属性,保存在 DPIS 中;文件本身元数据包括文件的属性如所有者、文件大小、最近访问时间、数据对象的位置等,保存在 MDS 中;数据对象包含文件的实际数据,保存在 OST 中。由于实际的数据块分配隐藏到 OST 中,并且文件可以包含若干个数据对象,因此,OCFS 的文件大小不再受限于传统设计的数据块个数,能够支持超大规模的文件以及超大容量的目录。用户访问文件首先通过 CFS 访问 DPIS 与 MDS,确定对文件的访问权限和获取文件的位置信息,然后通过网络与 OST 直接交换数据对象,数据对象的访问绕开了服务器的中转,能够实现高性能的并行数据传输。目录路径元数据、文件本身元数据以及数据对象的独立管理能够实现大规模集群存储系统的高可伸缩性。

1.2 元数据管理模型

区别于现有的元数据管理方法,在 OCFS 的元数据设计中,将文件的目录路径信息与元数据分开管理,这种设计能够避免或大大减少元数据的迁移,提高元数据的访问性能。

(1)目录路径 ID:在 OCFS 设计中引入目录路径索引服务器,能够为系统中的所有目录文件分配一个全局唯一的目录路径 ID。目录路径索引服务器采用共享存储的服务器集群,统一管理目录路径 ID 的分配,保证目录路径 ID 是全局唯一的。目录路径索引服务器中的索引项如下:INDEX = < DPID, DirectoryPath, AC_p >,其中,DPID 表示全局唯一的目录路径 ID,DirectoryPath 为目录路径名,AC_p 表示该目录的路径访问控制属性,路径访问控制属性的构造同现有的遍历文件路径中的所有目录来确定该文件的访问权限的方法相同,不同的是,每一次创建目录时将获得的路径访问控制属性记录在当前索引项中,并且在该目录下创建新的目录文件时,不再需要遍历前面目录,可以递归使用当前的路径访问控制属性与新建目录文件的访问控制属性构建新的路径访问控制属性。索引项由 DPID 唯一确定。这种设计使得文件路径中目录名和访问权限可以任意修改,DPID 始终保持不变,所以不会因为目录名和访问权限的修改导致该目录下所有文件的元数据更新,避免了大量的元数据迁移。特别约定根目录"/"的 DPID 为0。

(2)访问权限控制:现有的元数据管理方法中必须遍历文件路径中所有目录才能确定该文件的访问权限。在 OCFS 中,将文件本身的访问控制属性与其路径访问控制属性分开管理。路径访问控制属性由目录路径索引服务器管理,文件本身的访问控制属性保存在文件的元数据中。元数据服务器系统通过路径访问控制属性与文件本身的访问控制属性共同确定文件的访问权限,由于路径访问控制属性只需要一次散列计算就可以从目录路径索引服务器中获取,不再需要遍历全部目录,大大提高了访问性能。

(3)文件的位置信息:文件的位置信息由保存文件的数据对象的 OST 编号及该 OST 分配给文件的数据对象 ID 确定。定义文件的位置信息如下:LOCATION = < OST_NO, OID >,其中,OST_NO 表示保存文件数据对象的 OST 编号,OID 表示该 OST 分配给文件的数据对象 ID。

(4)元数据信息项:定义所有元数据的集合为 D,每一个文件的元数据信息项 $d \in D$,d 定义如下:ITEM = (DPID, Name, TYPE, AC, LOCATION, OTH-

ER),其中,DPID 表示文件的目录路径 ID,Name 表示全路径名中的最后一个分项名,TYPE 表示文件的类型,AC 表示文件的访问控制属性,LOCATION 表示文件的位置信息,OTHER 表示其他的元数据信息(文件大小、最近访问时间等)。DPID 在新建文件时获得。

元数据项在磁盘上的存储采取 B + 树算法,能够提供标准的树型层次结构,同一目录下的元数据项存储位置尽可能靠近,提高检索和存储效率。

1.3 元数据分配与定位算法

OCFS 的设计面向大规模的分布存储系统,为避免元数据的访问成为系统的访问瓶颈,OCFS 采用非共享存储服务器集群提供系统的元数据服务,集群中的每一个服务器是相互独立的,分别负责保存系统的一部分元数据信息,每一个服务器是自治的,不需要访问一个中央服务器,元数据服务器能够根据系统的负载情况自动的进行伸缩,并且对用户透明,始终保持线性的访问性能。由于元数据服务器的数量可能随着存储系统规模的变化而变化,因此 OCFS 不是简单的采用静态散列计算实现元数据的分配,而是采用动态线性散列计算实现元数据在不同元数据服务器之间均匀分配,散列计算需要唯一标示一个文件的关键字。由于 DPID 是全局唯一的,而同一个目录下文件名是唯一的,因此 OCFS 中依据文件的 DPID 与文件名作为标示该文件的关键字,对计算出来的关键字 KEY 进行散列计算分配到不同的元数据服务器。这种元数据分配算法既具有纯散列算法的优点,又可以有效避免或大大减少了元数据的更新与迁移。特别约定,根目录"/"的元数据分配到编号为 0 的元数据服务器。

1.4 访问权限控制

传统的元数据管理方法中,需要遍历文件所在的所有目录才能确定用户对该文件的访问权限。在 OCFS 中,将文件本身的访问控制属性与其路径访问控制属性分开管理。根据文件的 DPID 和文件名能够同时确定文件本身的访问控制属性 AC 与路径访问控制属性,带来的好处是,在获取文件元数据的同时,无需遍历文件所在的所有目录,即可确定用户对该文件的访问权限。

1.5 元数据更新

在根据散列计算分配元数据的方法中,不同文件的散列值不同,分配的元数据服务器也不相同,即使同一个目录下面的文件也完全可能分布到不同的元数据服务器中。典型的散列算法根据文件的全路径文件名进行计算,优点是能够直接定位文件的元数据信息,但是在修改目录(修改目录名、修改目录权限、删除目录)、增减集群中的服务器数量的情况下导致大量的元数据迁移。OCFS 中提出目录路径与元数据分离的设计能够避免或大大减少元数据的迁移。(1)修改目录名,OCFS 中散列计算的关键字只与文件名及 DPID 有关,与目录名、权限无关,因此只需要更新目录路径索引服务器中的索引项以及更新和迁移一条元数据信息(修改的当前目录文件);(2)修改目录权限,只需要更新目录路径索引服务器中的索引项,不需要更新和迁移元数据信息;(3)删除目录,只需一条消息(组播)即可;(4)增减集群中的服务器数量,在 OCFS 设计中,元数据服务器能够根据系统的负载情况自动的进行伸缩,并且对用户透明,始终保持线性的访问性能,即使在分裂的情况下,也只有一个服务器中的大约一半元数据迁移到新增服务器中,具有非常好的可伸缩性。

2 性能分析比较

表 1 将 OCFS 的元数据管理与以前的元数据管理方法进行了比较。

表 1 性能比较

设计方法	可伸缩性	平衡工作负载	访问性能	访问控制	元数据更新(修改目录名、权限,删除目录)
目录子树分割	差(需要人工操作)	较差(需要人工操作)	较差(需要遍历全路径)	较差(需要遍历全路径)	好(无需更新消息)
纯散列(全路径名)	较差(需要重新散列)	好(均衡分配)	好(直接定位)	较差(分不同情况)	差(需 $m*(n-1)/n$ 条消息)
混合	较差(需要重新散列)	好(均衡分配)	好(直接定位)	好(直接确定)	差(需 $m*(n-1)/n$ 条消息)
OCFS	好(自动伸缩)	好(均衡分配)	好(接近直接定位)	好(直接确定)	好(至多只需 1 条更新消息)

表1中：m 为该目录下的文件数目，n 为元数据服务器数目。

3 结 论

本文提出一种新颖元数据集群系统结构，并设计一种可伸缩的元数据管理模型，将元数据划分为目录路径元数据与文件本身元数据，由独立的目录路径索引服务器管理文件的目录路径名与路径访问控制属性，由元数据服务器管理文件本身的元数据。性能分析比较表明，该方法能够克服现有的元数据管理方法的缺点，能够有效平衡工作负载，避免因为目录修改而导致的大量元数据迁移，实现元数据服务系统的自动伸缩管理。

参考文献

[1] Ousterhout J K, Costa H D, Harrison D, et al. A trace-driven analysis of the Unix 4.2 BSD file system. In: Proceedings of the 10th ACM Symposium on Operating Systems Principles. Dec. 1985. 15

[2] Levy E, Silberschatz A. *ACM Computing Surveys*, 1990, 22 (4): 321

[3] Corbett P F, Feitelso D G. *ACM Transactions on Computer Systems*, 1996, 14(3):225

[4] Brandt S A, Miller E L, Long D D E, et al. Efficient metadata management in large distributed storage systems. In: the 17th International Parallel and Distributed Processing Symposium (IPDPS 2003), April 2003

[5] Braam P J. The Lustre storage architecture. Technical report, Cluster File Systems, Inc., 2002. http://www.lustre.org/docs/lustre.pdf

Design of a Scalable Metadata Cluster System

Liu Zhong, Wang Xiaodong, Zhou Xingming

(School of Computer Science, National University of Defense Technology, Changsha 410073)

Abstract

Directory subtree partitioning and pathname hashing are two common methods used for managing metadata, but both suffer from unbalancing workloads, large metadata migration and poor scalability. A novel metadata cluster system architecture and a scalable metadata management model are proposed, which divides metadata into directory path metadata and file metadata. Compared to traditional metadata management method, it can balance workloads efficiently, avoid large metadata migration and make the management of metadata service system be scaled automatically.

Key words: Distributed storage, Metadata cluster, Directory path, Scalable

可伸缩的元数据集群系统设计

作者： 刘仲， 王晓东， 周兴铭
作者单位： 国防科学技术大学计算机学院,长沙,410073