云数据存储与管理①

刘 琨，董龙江

(北京联合大学 应用科技学院，北京 102200)

摘 要：云计算作为一种新兴的商业模式发展异常迅猛，数据存储与管理是云计算中非常重要也极具价值的研究领域。介绍了云存储的概念、云存储的优势及云存储的架构；结合企业的具体实例，从多层次多方位深度剖析了云数据存储技术 GFS(Google File System)/HDFS(Hadoop Distributed File System)及云数据管理系统 BigTable/HBase，并对它们进行了分析比较。

关键词：云计算；云存储；数据管理；GFS；HDFS

Cloud Data Storage and Management

LIU Kun, DONG Long-Jiang

(College of Oriental Application & Technology, Beijing Union University, Beijing 102200, China)

Abstract: Cloud computing as a new business model is developed very fast. Data storage and management is a very important and valuable research field in cloud computing. This paper introduces the concept of cloud storage as well as the advantage and architecture of cloud storage. Then it analyzes the cloud data storage technology-- GFS(Google File System)/HDFS(Hadoop Distributed File System) and the cloud data management system--BigTable/HBase based on the specific cases of enterprises.

Keywords: cloud computing; cloud storage; data management; GFS; HDFS

近几年，云计算的概念越来越热门，美国 eWeek 网站评选出 2009 年 IT 界五大科技发展趋势，云计算位居首位。云存储是云计算中的核心研究领域，主要解决云计算中的数据存储与管理问题。目前，众多 IT 巨头们都在大力开发云存储技术及产品。例如，Google 一直致力于推广以 GFS[1]、BigTable[2]等技术为基础的应用引擎，为用户进行海量数据处理提供了手段。本文首先介绍了云存储的相关概念，然后结合企业的实例分析与讨论了云数据存储与管理技术。

1 云存储

1.1 云存储的定义

云存储指通过集群应用、网格技术或分布式文件系统等功能，将网络中大量各种不同类型的存储设备通过应用软件集合起来协同工作，共同对外提供数据存储和业务访问功能的一个系统[3]。云存储可以简单的理解为云计算中的存储，是配置了大容量存储空间的云计算系统。用户所有的数据都保存在“云”中，需要时从“云”中读取，本地不需要任何的存储设备。云存储更准精确地说是一种服务，用户使用的是由许多个存储设备和服务器所提供的数据访问服务。

1.2 云存储系统的架构

云存储系统主要用来进行数据存储与管理且处理的数据都是超大规模的，包括存储层、基础管理层、应用接口层和访问层。云存储的架构模型见图 1[3]。

存储层主要包括存储设备及存储设备管理系统。存储设备分布在不同地域，彼此之间通过网络互联在一起。存储设备管理系统负责存储设备的虚拟化管理、多链路冗余管理、硬件设备的状态监控和故障维护、设备升级等。

① 基金项目:北京联合大学校级科研项目(ZK2009606)
收稿时间:2010-10-13;收到修改稿时间:2010-11-16

基础管理层通过集群系统、分布式文件系统和网格计算等技术，实现云存储中多个存储设备之间的协同工作，同时负责对数据进行加密、备份、压缩等，保证数据的正确性与安全性，使云中的存储设备提供更强更好的数据访问性能。

应用接口层根据用户订购的服务为用户分配权限，为不同的用户提供不同的 API 接口及应用软件，同时提供网络接入、用户认证等功能。

访问层包括能够访问云存储系统的用户，用户可以通过标准的公共应用接口登录云存储系统，享受云存储服务。

目前，Google、Amazon、IBM 等各大公司都实现了自己的数据存储管理架构模型。本文将以 GFS[1]/BigTable[2]、HDFS[4]/HBase[5]为例分析云数据存储与管理技术。

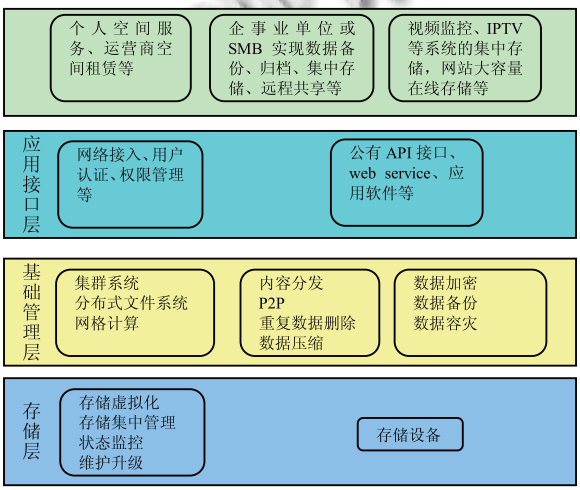

图 1 云存储模型

2 企业云数据存储技术

2.1 GFS

Google 为了满足迅速增长的数据处理需求，设计并实现了文件系统 GFS[1]。GFS 与传统的分布式文件系统有很多相同的设计目标，比如性能、可伸缩性、可靠性以及可用性。

2.1.1 系统架构

一个 GFS 集群包含一台主服务器（Master）、多台块服务器（ChunkServer）以及多个客户端，如图 2。所有的这些机器通常都是普通的 Linux 机器。

(1) 主服务器

主服务器主要负责管理文件系统的元数据（包括文件和块的名字空间、访问控制信息、文件与块的映射信息以及块副本的位置，通俗地说就是管理文件的目录结构）。也负责创建块及副本，回收不再使用的块空间，在块服务器间进行负载平衡等。

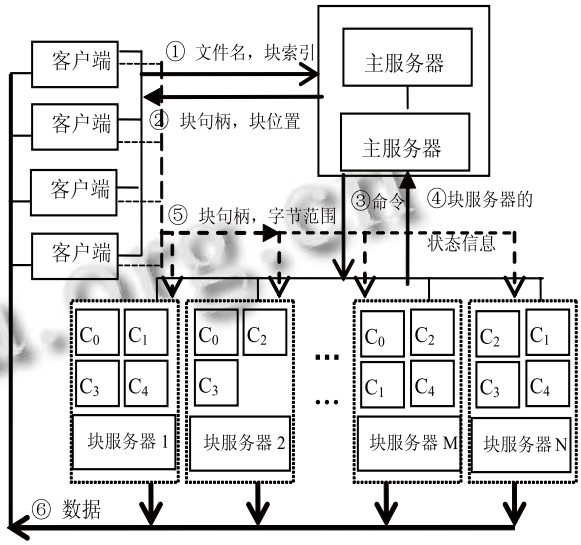

图 2 GFS 架构

GFS 采用的是单一的主服务器的策略，也就是说同一时刻只有一个主服务器提供服务，避免了为协调多台主服务器同步工作付出的代价。客户端与主服务器的交互只是获取元数据，其它所有数据操作都由客户端直接与块服务器进行通信，减少了对主服务器的读写，避免主服务器成为瓶颈。

(2) 块服务器

GFS 的文件被分割成固定大小的块，默认为 64M，存放在各个块服务器上。缺省每块复制到 3 个块服务器上，保存 3 个备份，用户可以为文件设定不同的复制级别。选用比较大的块尺寸的好处为：减少了主服务器需要保存的元数据量，客户端可以对一个块进行多次操作，减轻了网络负载；另一方面也有缺陷，如果块尺寸过大，由于小文件包含的块少，当多个客户端对同一个小文件进行多次访问时，存储这些块的块服务器就成为热点，根据反复实践确定为 64M。

在块创建的时候，主服务器为它分配一个不变的、全球唯一的 64 位的块标识。块服务器把块作为 Linux 文件保存在本地硬盘上，并且根据指定的块标识和字节范围来访问块数据。如图 2，显示了 4 个块服务器，其中有 5 个数据块 C0—C4，每个块都同时存放在 3 个服务器上。

(3) 客户端

客户端以类库的形式为用户提供了文件读写、目录操作等接口，当用户需要进行操作时，客户端配置相应接口。GFS 客户端代码实现了 Google 文件系统 API、应用程序与主服务器和块服务器通信、对数据的读写等功能，这些代码都嵌入到客户端的程序中。

2.1.2 工作流程

如图 2，图中细实线表示客户端与主服务器及主服务器与块服务器的控制消息，粗实线表示块服务器与客户端的数据通信，虚线表示客户端与块服务器的控制消息。客户端首先根据文件结构及块大小计算块索引；然后把文件名与块索引发送给主服务器（图中的标识①）；主服务器将块句柄、块副本的位置信息发送给客户端（图中的标识②）；客户端把块句柄及字节范围发送到最近的一个副本中（图中标识⑤）；块服务器返回块数据给客户端（图中标识⑥ ）。一旦客户端从主服务器中获得了块的位置信息后，客户端不再与主服务器进行交互（除非元数据信息过期或文件被重新打开），后续操作客户端直接与块服务器通信。

GFS 本地硬盘只存放文件的目录结构和分块信息，而块的位置信息则是实时计算的。主服务器不永久保存块服务器与块的映射信息，而是对块服务器轮询来获得这些信息。当主服务器启动或者有新的块服务器时，主服务器向各个块服务器轮询从而获取块信息（图中标识③④）。主服务器也周期地与每个块服务器通信，发送指令到各个块服务器并接收块服务器的信息（图中标识③④）。

GFS 还提供快照和记录追加操作功能。快照可以瞬间对一个文件或者目录树做拷贝，并且不会对正在进行的其它操作造成任何干扰。记录追加在保证追加操作的原子性的条件下允许多个客户端同时往一个文件追加数据。这样多个客户端可以在不加附加锁的情况下同时追加数据。

2.1.3 容错

(1) 块服务器。当某个块服务器不能正常工作时，如果主服务器没有发现，仍然把这个块服务器分配给客户端，势必会造成用户无法得到所要的信息。所以主服务器要时刻了解块服务器的状态，通过周期性的心跳信息监控块服务器的状态来保证它持有的信息是最新的。

(2) 主服务器。 GFS 使用数据库中日志的原理处理主服务器崩溃的情况。名字空间、文件与块的映射信息会记录在系统日志文件中，日志文件存储在本地硬盘上并复制到其它远程主服务器上，这样当主服务器崩溃时数据也不会丢失。操作日志包含了关键的元数据变更历史记录。当主服务器崩溃时，通过重演操作日志把文件系统恢复到最新的状态。当操作日志增长到一定值时主服务器对系统状态做一个镜像，并将所有的状态数据写入镜像文件，此时旧的镜像文件及日志可以被删除。在系统恢复的时候，主服务器读取这个镜像文件，根据镜像文件及最新的日志文件恢复整个文件系统。

(3) 客户端。若某个客户端正在写文件时却不能正常工作，其它客户端也将无法访问这个文件。GFS 使用租约机制解决这个问题[6]。当客户端要占有某个文件时，与主服务器签订一个租约，初始设定为 60 秒，当块被修改后，主块可以申请更长的租约，租约申请信息及批准信息在主服务器与块服务器的心跳消息中传递。如果某个客户端崩溃了，当租期到期后，主服务器可以把此文件分配给其它客户端。

2.2 **HDFS**

2.2.1 Hadoop 简介

Hadoop 是 Doug Cutting-- Apache Lucene 创始人开发的使用广泛的文本搜索库，起源于开源的网络搜索引擎 Apache Nutch。Hadoop 最出名的是 MapReduce 及其分布式文件系统 HDFS，同时还有很多子项目提供补充性服务，详细内容见参考文献[7]

2.2.2 DFS 简述

HDFS 全称为 Hadoop Distributed File System ，它是 Hadoop 的一个子项目，运行于大型商用机集群，基本是按照 Google 的 GFS 的架构来实现的。HDFS 采用 master/slave 架构。一个 HDFS 集群有一个 Namenode 和多个 Datanode 组成。Namenode 是中心服务器，相当于 GFS 中的 Master，负责执行文件系统的命名空间操作，如打开、关闭、重命名文件和目录，同时决定块到 Datanode 节点的映射。Datanode 相当于 GFS 中的 ChunkServer，负责管理节点上的存储，进行块的创建、删除和复制等。HDFS 中一个文件分成一个或多个块（block），这些块存储在 Datanode 集合里。Namenode 和 Datanode 都设计成可以运行在普通廉价的 Linux 机器上。HDFS 采用 java 语言开发，因此可以部署在很大范围的机器上[4]。如图 3 为 HDFS 的架

构模型。

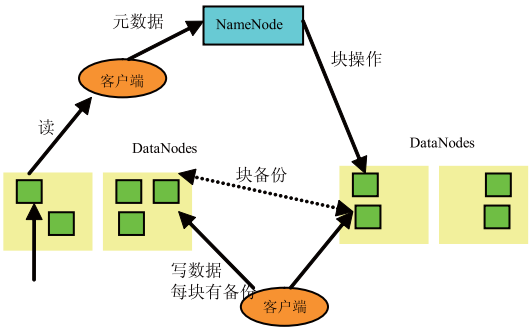

图 3 HDFS 架构

HDFS 与 GFS 很多原理都相似，这里不再赘述。但它们也有不同之处，例如 HDFS 缺少快照和记录追加操作，同时也不支持并行写；数据一致性方面，HDFS 更简单，对于失败的写，结果显示为“不一致”，成功的为“已定义”；系统交互方面，DataNode 基本不处理租约；主服务器上的操作，HDFS 也比较简单，它不区分读/写锁；垃圾回收上，HDFS 目前并没有实现回收站的功能。总的来说，HDFS 基本实现了 GFS 的一些目标，但还有很多的功能需要实现[8]。

3 企业云数据管理系统

3.1 BigTable

3.1.1 实例分析

BigTable[2]是 Google 公司设计的用来处理海量数据的分布式结构化数据存储系统，它处理的数据通常是分布在数千台普通服务器上的 PB 级的数据。BigTable 的实现满足了以下几个特性：适用性广泛、可扩展、高性能和高可用性，它已经在超过 60 个 Google 的产品和项目上得到了应用，例如 Google Analytics、Google Finance、Orkut、Personalized Search、Writely 和 Google Earth。

BigTable 的三个基本要素为：行（row）、列族(column families)、时间戳(timestamps)。如图 4，在 BigTable 中保存域名为 www.cnn.com 的页面。行名是 URL 的反向，列族 contents 用来保存网页内容，列族 anchor 保存引用该网页的锚链接文本，由于 cnn 的主页被“Sports Illustrater”和“MY-look”的主页引用，因此包含了两列”anchor:cnnsi.com”、”anchor:my.look.ca”。时间戳表示每个列的版本。每个锚链接只有一个版本，所以只有一个时间戳，分别为 t7 与 t8；页面有三个版本，所以 contents 列族包括三个时间戳分别为 t1、t2、t3。可以定义更多的列族。

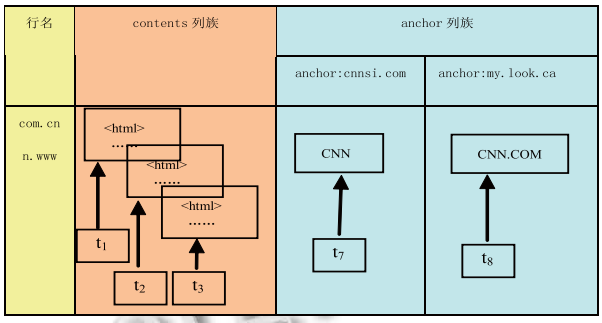

图 4 Web 页面示例

3.1.2 实现原理

• 行。行关键字可以是任意的字符串，目前最多为 64K， 所有行按照字典顺序排序。对同一个行关键字的读或写操作都是原子的，用户访问数据时给一行或者几行数据加锁。

• 列族。列关键字组成的集合叫做“列族”，同一列族存放相同的类型的数据。列族必须先创建才能存放数据。一张表中的列族最多几百个，并且列族在运行期间很少改变。

• 列。列关键字的命名语法为：列族：限定词。列族的名字必须是可打印的字符串，而限定词的名字可以是任意的字符串。例如上例中 anchor 为列族，cnnsi.com 与 my.look.ca 都是限定词。再如，也可以有列族 language，用来存放撰写网页的语言。在 language 列族中只使用一个列关键字，用来存放每个网页的语言标识 ID。

• 时间戳。由于每一个数据项都可以有不同的版本（比如网页经常更新时就会产生不同版本），不同版本的数据通过时间戳来索引。时间戳的类型是 64 位整型。时间戳可以由 BigTable 或者用户程序赋值，如果用户不定义则使用当前时间。不同版本的数据按照时间戳倒序排序，即最新的数据排在最前面，用户可以指定只保存最后 n 个版本的数据。

• Tablet。BigTable 又划分出为多个 Tablet,每个 Tablet 包括多个行，每个行可以动态分区。一个 Tablet 大概有 100-200 MB，每个机器存储 100 个左右的 Tablet。

• SSTable。BigTable 使用 GFS 存储日志文件和数据文件。存储文件的格式为 Google SSTable。SSTable

使用关键字（key）到值（value）映射的数据结构，关键字和值都是任意的字符串。SSTable 划分为多个数据块（数据块大小可以配置，典型配置为 64K），使用块索引（通常存储在 SSTable 的最后）来定位数据块。在打开 SSTable 的时候，索引被加载到内存，每次查找时先用二分查找法在内存中的索引里找到数据块的位置，然后再从硬盘读取相应的数据块。也可以选择把整个 SSTable 都放在内存中，这样就不必访问硬盘了。

• Chubby。BigTable 采用了一个分布式锁服务组件 Chubby[9]。一个 Chubby 服务包括 5 个活动的副本，其中的一个副本被作为 Master，只有在大多数副本都是正常运行的，并且能够互相通信的时，Chubby 服务才是可用的。

3.2 HBase

HBase[5,7,10]是 Hadoop 的正式子项目，它是一个面向列的分布式数据库，其思想源于 Google 的 BigTable。HBase 的索引是行关键字(row key)、列关键字(column key)和时间戳(timestamp)。表是疏松存储的，用户可以给行定义各种不同的列。所有行都按照词典顺序排序，每行由一个可排序的主键和任意数量的列构成。HBase 做写操作时，每一行都是一个原子元素，用户访问数据时给一行或者几行数据加锁。

列名的格式为<family>：<label>，family 必须是可打印的字符串，label 是任意的字符串。每个表的 family 集合是固定不变的，只能通过改变表结构来改变，label 的值是可以改变的。HBase 要求列族的个数小于 100。HBase 时间戳的概念类似于 BigTable。HBase 分为多个 regions，类似于 BigTable 的 tablet，Region 大小是可配置的，默认为 256MB。

HBase 可以使用任何文件系统，只要有该文件系统的代理或者驱动即可，例如 HDFS、S3、 S3N、EBS。内部存储数据的文件的格式为 Hfile，其中数据块的大小是可配置的，典型配置是 64K，使用块索引来定位数据块。HBase 不支持存储文件到内存的映射。ZooKeeper 被 HBase 用来协调任务并非当成锁服务。HBase 使用 ZooKeeper 达到了 BigTable 使用 Chubby 的效果。HBase 支持多个 Master。

4 云存储的发展

我国对云存储技术的研究处于刚刚起步阶段，但云存储已经成为未来存储方式发展的一种趋势。在《中国云存储服务报告，ChinaCloudStorageServicesReport》中指出未来 5 年，中国云存储服务市场的年复合增长率将达到 103%，中型企业将成为中国第一轮大规模采用云存储服务的企业。也有机构预测，到 2013 年，企业对私有云的投资会超过公有云，至少 5:1。在云存储时代，目前各个云存储企业及研究机构也正在将各类搜索、虚拟化等技术与云存储相结合，从而能够提供一系列的数据服务。云存储相比普通的存储技术有许多的优势[11]：

(1) 减少企业投入成本。普通企业需要投资很多才能构建自己的数据中心，而云存储服务商有专业的存储解决方案，所以对于普通企业租用公共云存储更合适。

(2) 能够很好的应付突发的大访问量。普通企业的数据中心通常不能应对突发性的大访问量，比如大型赛事的购票系统，但云存储使用了服务器集群和虚拟化技术，可以临时调用集群中的各个设备。

(3) 能够提供存储服务的同步升级和数据的有效管理。如果企业自己构建数据中心，需要购买各种设备管理软件，负责设备和软件的升级、维护及管理。使用云存储服务，则可以把这些工作交给专业的云存储服务商来进行。所以说云存储是未来存储发展的一种趋势。

但是，云存储的发展也面临很多问题，这些问题不解决势必会影响云存储技术的发展及推广应用。

① 安全性。由于数据存储在云中，各个用户都能访问，因此保证数据的安全是首要问题。数据加密技术、数据备份等技术的应用保证了数据的安全性。

② 网络带宽。由于云的服务器及用户分布在网络中的各个地方，所有的数据都需要在网络中传输。目前基本上是通过 ADSL、DDN 等宽带接入设备的，只有带宽充足了，才能提高传输速度，用户才能更好的享受云存储的服务。

③ 数据管理。由于云服务器是各个云厂商提供的，分布广泛且配置不同。当用户需要访问数据时，应该能够快速地找到，当用户存储数据时，应该能够把数据存放在合适的服务器中，而且必须解决服务器的故障等问题。这些都需要进行管理。

④ 云数据中心的建设及维护问题。建设云数据中心需要大量的资金投入，对于我国国内企业来说还是一个很大的挑战，虽然国内建设了部分的云数据中心，

但由于用户少，维护一个云数据中心也是一个挑战。

这些只是我国云存储发展处于起步阶段面临的问题，随着更多的厂商的加入及用户的使用此问题便会迎刃而解。

5 结语

云计算是互联网发展的必然产物，它的出现也为互联网带来了更丰富的应用。云数据存储技术及数据管理技术是云计算中的核心领域，主要解决了在“云”这个大环境中的数据存储及管理模式。本文主要讨论了云存储的概念、优势及架构，分析了 GFS、HDFS 数据存储技术以及 BigTable、HBase 数据管理系统。虽然目前云存储处于起步阶段，也面临很多困难，但随着更多企业及学术界对云存储的研究，云存储技术会给我们的生活带来更多的便捷，云存储技术也会更加地成熟。

参考文献

1 Ghemawat S, Gobioff H, Leung ST. The Google file system. Michael L. Scott, ed. Proc. of the 19th ACM Symposium on Operating Systems Principles. New York: ACM Press, 2003: 29–43.

2 Chang F, Dean J, et al. Bigtable:A Distributed Storage System for Structured Data. ACM Trans. on Computer Systems, 2008,26(2):1–26.

3 中国云计算网.什么是云存储. (2008-11-17)[2010-08-25]. http://www.cloudcomputing-china.cn/Article/luilan/200811/215.html

4 Borthakur D. The Hadoop Distributed File System: Architecture and Design. (2008-09-02) [2010-08-25].http://hadoop.apache.org/common/docs/r0.16.0/hdfs_design.html

5 Hbase Development Team.HBase: Bigtable-like structured storage for Hadoop HDFS.(2010-08-10)[2010-08-25]. http://wiki.apache.org/hadoop/Hbase.

6 分布式基础学习.(2009-2-22)[2010-8-25]. http://www.cnblogs.com/duguguiyu/archive/2009/02/22/1396034.html.

7 White T. Hadoop:The Definitive Guide. California: O’Reilly Media,Inc. 2009:12–14.

8 Caibinbupt. Hadoop 源代码分析(重读 GFS 的文章). (2009-01-29)[2010-8-25].http://caibinbupt.javaeye.com/blog/318949.

9 Burrows M. The chubby lock service for looselycoupled distributed systems. Brian Bershad,ed. Proc. of the 7th USENIX Symposium on Operating Systems Design and Implementation. New York: ACM Press, 2006:30–40.

10 吴吉义,傅建庆,张明西,等.云数据管理研究综述.电信科学,2010,26(5):34–41.

11 李煜民,章才能,谢杰.云计算环境下的数据存储.电脑知识与技术,2010,6(5):1032–1034.

(上接第 231 页)

21(6):706–718.

8 Li CF Guo GC. Progress in quantum information research. Progress in Physics, 2000,20(4): 407–431 (in Chinese).

9 首次在国际上实现量子分解算法.中国科学院院刊,2008,23(1):76–76.

10 彭卫丰,孙力.SHOR 量子算法的优化及应用研究.计算机应用与软件,2009,26(5):239–246.

11 Grover Lov K. A fast quantum mechanical algorithm for database search. Proc. of the 28th Annual ACM Symposium on the Theory of Computing, 1996.

12 Grover LK. A Framework for Fast Quantum Mechanical Algorithms. Proc. of the 30th Annual ACM Symposium on Theory of Computing, 1998.

13 孙吉贵,何雨果.量子搜索算法.软件学报,2003,14(3):334–344.

14 李士勇,李盼池.量子计算与量子优化算法.哈尔滨:哈尔滨工业大学出版社,2009.

15 Han KH, Kim J H. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Trans on Evolutionary Computation, 2002,6(6): 580–593.

16 王凌.量子进化算法研究进展.控制与决策,2008,23(12): 1322–1326.