一种基于 IP 网络的统一对象存储访问模型的研究*)

刘 群 冯 丹 覃灵军 曾令仿

(华中科技大学计算机学院 信息存储系统教育部重点实验室 武汉 430074)

摘 要 随着计算机应用的普及和存储需求的膨胀,存储系统正在由 NAS 和 SAN 转向了 OBS。本文介绍了三种存储系统的访问模式,在 OBS 基础上,提出了存储设备对象,这不仅丰富了对象的内涵,还总结出一种统一对象存储访问模型,对不同存储对象设备实行访问协议的映射,使之更具有普适性。

关键词 基于对象存储,存储设备对象,统一对象,存储访问模型

Research for a Compatible Object Storage Access Model Based on IP Network

LIU Qun FENG Dan QIN Ling-Jun ZENG Ling-Fang

(Key Laboratory of Data Storage System, Ministry of Education

School of Computer, Huazhong University of Science and Technology, Wuhan 430074)

Abstract With computer application popularizing and storage capability expanding, research for NAS and SAN storage system are changing for OBS. This paper describes access pattern for three storage systems and gives storage device object based on OBS. It can not only enrich the concept of object, but also generalize a compatible object storage access model. This model maps access protocol for different storage device object. And it is more universalizable.

Keywords Object-based storage, Storage device object, Compatible object, Storage access model

1 引言

随着计算机技术的发展和应用的普及,信息存储容量成爆炸性地增长,传统的直接存储(Direct Access Storage, DAS)模式已经显得非常力不从心。特别是网络技术的崛起,在富有挑战性课题研究的带动下,存储模式从以服务器为中心转向以数据为中心的网络存储模式,典型代表是附网存储(Network Attached Storage, NAS)和存储区域网(Storage Area Network, SAN)[1]。

NAS 采用"文件"作为接口,通过网络接口把存储设备直接连入到网络中,支持 NFS 和 CIFS 的网络文件协议,并具有易用性和可管理性,实现细粒度数据共享以及跨平台文件共享,但系统延迟大,访问路径将可能成为系统中瓶颈;SAN 采用"块"作为接口,是用于数据存储的高速专用网络,通过专用平台向外界提供服务,提供内部任意节点间多路可选择的数据交换,更方便地共享存储设备,但数据共享粒度大,元数据服务器有可能成为系统的瓶颈。

基于对象存储(Object Based Storage, OBS)技术是以数据为中心网络存储模式。OBS 技术克服了 NAS 与 SAN 中不足,采用了对象作为接口,它既有"块"接口的快速,又有"文件"接口的便于共享[2]。对象使文件和存储元数据管理进行隔离,突破了 SAN 的文件共享限制和 NAS 系统中常见的数据路径瓶颈,使之在安全性、跨平台数据共享、高性能和可扩展性特性中更胜一筹[3]。

在访问模式上,NAS 和 SAN 仍然采用传统的应用—文件—块,只是文件和块所访问之处不同(如图 1 所示)。在 OBS 中,以文件数据访问为例,采用了应用—文件—对象—块模式。本文在 OBS 技术上提出存储设备对象,概括出统一对象存储访问模型,该模型提供统一对象接口,可适应于多种形式的存储对象设备,如 RAID、磁带、NAS、OBSD 等设备。

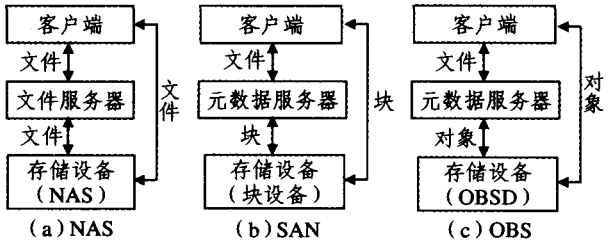

图 1 三种存储系统访问模式

2 基于对象存储

在 OBS 中,与固定大小的块不同,对象是可变长的,可包含任何类型的数据,如文件、数据库记录、医学图像以及多媒体视频音频等,至于包含何种类型数据由应用决定,对象还可动态地扩大和缩小。对象还具有描述其特征的属性,如对象的创建时间、访问时间、网络带宽、延迟时间等。

基于对象存储设备(Object-Based Storage Device, OBSD)维护所有对象的空间分配和所有空闲空间管理有关的元数据。在 OBSD 中,对象的存储建立在一个"平坦"文件系统基础上,实现一个平坦的一维空间[4]。

OBS 一个明显的好处就是把存储空间分配交由 OBSD 管理。在传统的基于块的文件系统中,其由用户部分和存储部分组成,OBS 将用户部分不变,而存储部分下移到 OBSD 中,相应的设备接口也从基于块的接口变为基于对象接口。由此文件系统的上层只负责把文件名等逻辑名称映射为对象 ID,而 OBSD 管理着存储空间的分配与数据组织,实现扁平目

*)基金资助:973 计划项目"下一代互联网信息存储的组织模式和核心技术研究"(2004CB318201),国家重大基础研究前期研究专项"网络存储数据行为的基础理论研究"(2004CCA07400)。刘 群 博士生,研究方向为计算机系统结构与高性能网络存储;冯 丹 教授,博导,主要研究方向计算机系统结构、磁盘阵列技术与高性能网络存储;覃灵军 博士生,研究方向为计算机系统结构与高性能网络存储;曾令仿 博士生,研究方向为计算机系统结构与高性能网络存储。

录将对象 ID 映射到存储块设备上。

根据 T10/SCSI OSD-2 草案协议[5]，对象可分为四种：根对象(Root Object)，分区对象(Partition Object)，集合对象(Collection Object)和用户对象(User Object)。根对象为 OBSD 中逻辑单元 OSD 的对象，且每个 OSD 只有一个根对象，其属性描述了该 OSD 的特性，如总容量、最多可以存储的分区对象数以及与数据完整性有关的属性等；分区对象包含了一组集合对象和用户对象，其属性包括了分区内的用户对象数目，分区内用户对象占用的空间等；集合对象用于实现用户对象的快速检索，一个分区对象可以包含零个或多个集合对象，一个用户对象可以属于零个或多个集合对象；用户对象最多，如文件、数据库记录存放在用户对象，其属性包括数据的创建、修改等。所有对象以 Partition_ID 和 User_ID 标识。此外，对象的属性存放在 OSD 上，使得不同存储应用之间的数据共享变得很容易，同时 OSD 集成了处理能力，使之成为了一种智能设备，为存储设备的自管理提供了机会。OSD 能理解设备中块之间的一些关系，并能利用这些信息，优化数据布局，预读取和缓存应用数据组织数据。

3 基于存储设备对象的存储系统

3.1 存储设备对象

存储设备对象，即数据存储的基本单位，它是基于存储设备对象的存储系统中的关键。其与 OBSD 不同之处是本身具有“接口”与“状态”标识，由 128bit 唯一对象 ID(UOID)识别，并由数据、属性和方法组成，这样扩展了对象内涵，将形形色色的存储设备(如 NAS、RAID 等)接纳其中，它最显著特点就是兼容性与可扩展性，能将各种异构存储设备、各种类型的应用数据、各种对象的属性与方法等可动态地扩大和缩小。

对于属性，都是以存储设备对象为基础，定义该对象所具有的特性，它包含对象的静态信息和动态信息，暗示了对象的行为。为了属性设置的方便和标准的统一，我们对 SCSI OSD-2 草案协议中属性进行扩展，规定存储设备的属性页码为 C000 0000h 到 EFFF FFFFh，即 D=C000 0000h，例如 D+5h 表示 C000 0005h，同时定义存储设备属性页(见表 1)，并且增加了许多与存储设备对象相关的属性，如读/写速率、访问模式(串行或随机)、生存时间等，由于篇幅有限，在此略去具体存储设备对象属性信息。

表 1 定义存储设备对象属性页

Page Number	Page Name	Page Format Defined	Support Requirements
D+0h	Device Directory	No	Mandatory
D+1h	Device Information	No	Mandatory
D+2h	Device Quotas	Yes	Mandatory
D+3h	Device Timestamps	Yes	Mandatory
D+4h	Reserved		
D+5h	Device Policy/ Security	Yes	Mandatory
D+6h to D+7Fh	Reserved		

同时也扩充了方法以及方法的描述，派生出新类型的对象命令，如 ADD、DELETE 等，同时可根据应用需求进行调整和扩充。

3.2 基于存储设备对象的存储系统

基于存储设备对象的存储系统(Storage System Based on Device Object，SSBDO)如图 2 所示，它是由元数据服务器(Metadata Server，MDS)、存储设备对象(Storage Device Object，SDO)、客户端(Client)以及高速网络组成。SSBDO 实现了异构存储设备对象的分层管理、数据分条存储、易扩展的存储系统。

MDS 负责提供全局命名空间，驻存着各种存储设备对象的基本信息，通过客户检索系统中的存储设备对象信息，根据存储设备的接口与状态，管理逻辑请求(如文件)到存储设备对象的映射，并提供身份认证等一系列安全机制。

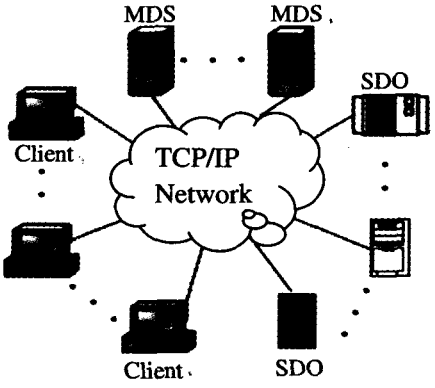

图 2 基于存储设备对象的存储系统

存储设备对象 SDO 包括处理器(CPU)、内存(Memory)、网络接口(NIC)和设备接口(Device Drive)[4]。SDO 可以多种形式存在，可以是 RAID；可以是 NAS；可以是磁带设备；可以是 OBSD 等。依照 SDO 的接口与状态，实现多种数据组织的存储。SDO 与传统的存储设备区别不是介质，而是接口，并且在 OBS 基础上进一步丰富和扩展了对象接口的内涵，使之成为一个通用存储访问的接口模式。

以文件数据请求为例，当 Client 向 MDS 发出独立的文件访问请求，MDS 返回给 Client 一张缓存对象 map 表和安全访问认证书，并利用它访问保存在 SDO 上的数据对象。一旦 Client 得到这张 map 后，直接与 SDO 交互。SDO 接受 Client 请求后，通过安全访问管理，执行对象访问相应的操作方法[6]。

4 统一存储访问模型

对于对象而言，其综合“文件”和“块”二者的优点，同“块”类似，对象是逻辑存储单元，能够不通过服务器直接对数据进行访问，同“块”一样提供性能上的优势；与文件相似，对象通过元数据服务器尽可能的抽象的东西进行访问，有细粒度的共享[6]和跨平台共享。

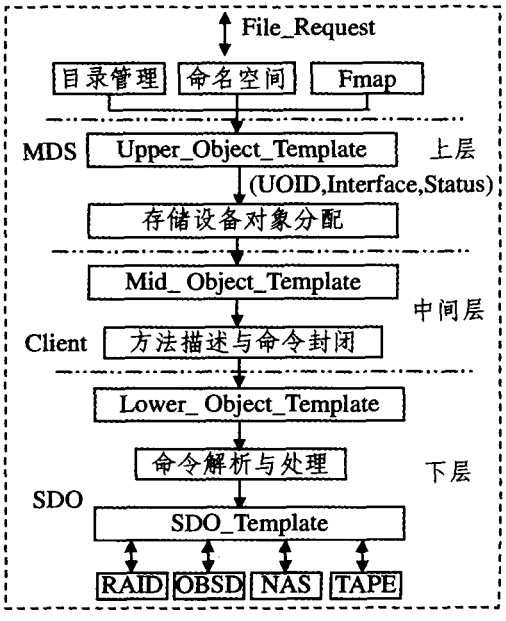

图 3 统一对象存储访问模型

根据对象访问的特性，提出了统一对象存储访问模型(如图3所示)。在SSBDO中，统一对象访问模型为三层，上层为Upper_Object_Template，依据存储设备对象接口、状态以及属性，以提供满足存储需求的存储设备对象，它由MDS负责接收请求，并把其转化为对象请求接口，然后交给存储对象中间层来处理。对象中间层Mid_Object_Template由Client负责，填充对象方法的描述，并将其转化为具体实施对象，中层是统一对象访问模型的核心，关系到将如何具体实施上层的对象请求与实现。对象下层Lower_Object_Template则是接收来自中间层的对象具体服务，根据不同的服务方法，由SDO完成相应的功能，它不仅需定义内部接口和服务，还为对象分配存储空间。

由上述可知，不论是SCSI协议、RAID协议，还是SCSI OSD-2草案协议，都可采用统一对象访问模型，它们均保留了对象模型中的上层和下层，仅仅修改具体实施方法。这样，不论OSD为何种类型的设备，基于对象存储系统中统一的存储对象模型。

结束语 本文通过分析三种网络存储系统的访问模式，在基于对象存储的基础上提出了统一对象存储访问模型，它给出了网络存储系统中一般性接口的定义。不论随机存储还是顺序存储，不论是硬盘还是阵列，不论是NAS还是OBSD，对每一种存储设备对象都有其相对应的接口，使其更具有普适性。

参考文献

1 Mesnier M, Ganger G R, Riedel E. Object-based storage. Communications Magazine, IEEE, Aug. 2003, 41(8): 84～90
2 Intel Corporation. Object-Based Storage: The Next Wave of Storage Technology and Devices. January 2004, accessible from http://www.intel.com/labs/storage/osd
3 Panasas white paper, Object Storage Architecture: Defining a new generation of storage systems built on distributed, intelligent storage devices. http//www.panasas.com. Sep. 2003
4 Feng Dan, Qin Ling-jun, Zeng Ling-Fang, Liu Qun. A Scalable Object-based Intelligent Storage Device [C]. In: Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004, 387～391
5 T10/1731-D Revision 0. SCSI Object-Based Storage Device Commands -2(OSD-2). http://www.t10.org. October 2004
6 Azagury A, Dreizin V, Factor M, et al. Towards an Object Store. In: Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies(MSS'03), 2003

(上接第81页)

来的关于信息比特$u(k)$的对数似然值。同样，对于译码器2，$Le12(u(k))$为外信息，即译码器1传递过来的关于信息比特$u(k)$的对数似然值，Turbo译码就是通过在两个译码器之间相互交换外信息，迭代译码。

在以往的迭代译码过程中，由于仅接收到受干扰的信息，得不到信息的似然值，一般，我们认为信息比特取0与1的概率相等，则$Le21(u(k))=0$。即第一次迭代译码的外信息我们设置为0。而通过软解调，我们得到传输比特的软信息，即传输信息的对数似然比，在第一次迭代译码时，可以将其视为MAP译码器1的先验信息。即

$$Le21(u(k))=l(u(k)) \tag{11}$$

3 性能仿真

我们采用如图1所示的仿真原理框图，对高阶调制的Turbo译码进行了性能仿真，采用加性高斯白噪声信道。Turbo码的成员编码器采用3个寄存器的(13,15)RSC编码器，编码长度为N=417，编码效率为R=1/3。采用16QAM的调制方式，解调采用式(7)所示的简化软解调算法，Turbo译码采用MAP译码方式，迭代次数为4次。其性能仿真曲线如图5所示。

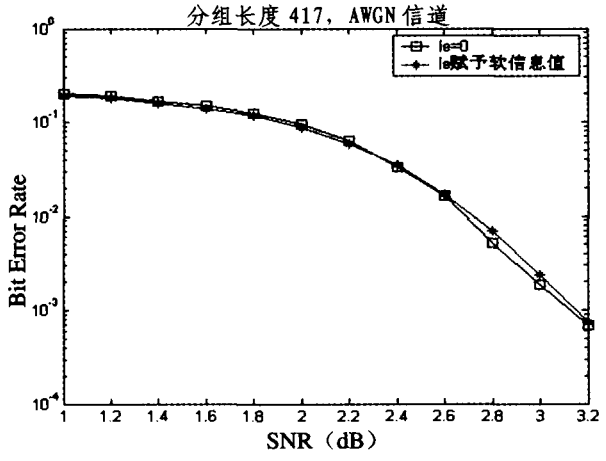

图5 分组长度N=417，R=1/3，16QAM调制方式的性能曲线图

结论 可以看到，在低信噪比的时候，采用(11)式对译码器1赋值，可以提高译码器的性能，而在信道环境较好时，未预先赋值的译码性能比较好，通过比较可以看到：在低信噪比时，将软解调信息作为先验信息参于迭代译码能带来0.1dB的增益，而在高信噪比时，不采用软解调信息比采用解调信息有0.1dB的增益。在低信噪比时，由于信道对信息的影响较大，信息的失真较大，采用解调的信息作为先验信息对传输信息进行了有益的补充，可以提高系统的性能，而在信道环境较好时，传输信息受到的干扰很小，采用软解调信息作为先验信息会干扰译码器的译码。因此，在系统中，我们可以采用一种自适应的调制与译码方式，在信噪比较低时，采用软解调信息补偿的迭代译码方法，而在高信噪比时，不进行信息的补偿。采用这种自适应的方法能够提高高阶调制的Turbo码的译码性能。

参考文献

1 Berrou C, Thitimajshima P. Near Shannon limit error correcting coding and decoding: Turbo codes [A]. In: Proc. ICC'93 [C]. Geneva, Switzerland: ICC, 1993. 1064～1070
2 Benedetto S, Montorsi G. Unveiling, Turbo codes: Some results on parallel concatenated coding schemes. IEEE Trans. Information Theory, Sept. 1995
3 Benedetto S, Montorsi G. Design of Parallel concatenated convolutional codes. IEEE Trans. Commun., 1996, 44: 591～600
4 Ungerboeck G. Channel coding with multilevel/phase signals [J]. IEEE Trans Inform Theory. 1982, T-28: 55～67
5 Robertson P, Worz T. A novel bandwidth efficient coding scheme employing Turbo codes [A]. In: Proc. ICC'96 [C]. 1996. 962～967
6 Robertson P, Worz T. Bandwidth-efficient turbo trellis-coded modulation using punctured component codes. IEEE Journal on Selected Areas in Communication, 1998, 16(2): 206～218
7 Goff S L, Glavieux A, Berrou C. Turbo-codes and high spectral efficiency modulation [A]. In: Proc. ICC'94 [C]. 1994. 645～649
8 Viterbi A J, Jack K, Wolf, et al. A pragmatic approach to trellis-coded modulation [J]. IEEE Communication Magazine, 1989, 27(7): 11～19