文章编号:1001 - 9081(2005)12 - 2820 - 04

高可靠性分布式虚拟存储系统的研究

曹 楠,康慕宁
(西北工业大学 计算机学院,陕西 西安 710072)
(GlassesWorm@yahoo.com.cn)

摘 要:针对分布式存储安全性及可靠性方面的一系列问题,提出了全新的分布式存储解决方案 RayStorage 系统,阐述了该系统的核心分布式虚拟存储架构(Distributed Virtual Storage Architecture, DVSA)的主要思想,并重点给出了 DVSA 中有关存储模式的若干概念及其效率分析结果。

关键词:分布式系统;虚拟网络存储;存储模式
中图分类号: TP309.3 **文献标识码**:A

Study on a high reliable distributed virtual storage system

CAO Nan, KANG Mu-ning
(*College of Computer, Northwestern Polytechnical University, Xi'an Shaanxi* 710072, *China*)

Abstract: Aiming at the security and stability problems in distribute storage, a new distributed storage solution called RayStorage system was presented, and the main notion about "Distributed Visual Storage Architecture"(DVSA) which is the kernel of the ReyStorage System was discussed, and the basic concepts of storage pattern and the results of their efficiency analysis was given out.

Key words: distribute system; virtual network storage; storage pattern

0 引言

随着计算机技术的发展,存储设备容量不断增长,在带来数据存储便利的同时,有研究表明[1],在 Windows2000 系统中,存储设备所拥有的资源并没有得到充分的利用,并且这种趋势随着存储设备容量的不断增长而愈加严重。同时,网络技术给存储界带来了全新的变革,由于本地数据传输速率与网络传输速率之间差异正在逐渐缩小,数据存储正在经历着由本地化到网络化的巨大转变。因此,如何利用分布于网络中各个异构主机节点之上的不可靠空闲存储资源来实现分布式的网络存储具有非常重要的研究与应用价值。

本文所提出的 RayStorage 系统是一种应用在 Internet 范围内的分布式虚拟存储系统,系统运行在一系列分布的、不可靠的,并且平台异构的主机节点之上,通过定制架构在主机级虚拟存储概念之上的虚拟存储服务,实现了对网络存储资源的管理、分配与利用。由于存储资源自身的分散性、异构性及不可靠性使得如何为用户提供安全可靠的服务成为系统研究的核心。虚拟存储[2]作为下一代网络存储的发展趋势,具有非常广阔的内涵与外延,其中主机级的虚拟存储,屏蔽了一切存储设备硬件之上的差异,具有很强的兼容性及可扩展性,相关领域的基本概念及成功应用为 RayStorage 的研究与设计提供了宝贵的经验及参考,同时也为 RayStorage 奠定了理论基础。

RayStorage 不同于现有的集中式的网络存储解决方案,例如 NAS、SAN[3]等,它提供了更为广阔、更加安全的数据复制机制,避免了系统单边崩溃的隐患;也不同于现有的大多数分布式存储系统,例如 RDSS[4,11]、OceanStore[5]等,RayStorage 提供了多层次的数据存储与访问机制,既能够进行上层存储模式级的数据操作,也能够实现安全高效的底层 Block 级的存储控制;虚拟存储相关概念的引入,也使得 RayStorage 系统具有更好的可控性及扩展性。

RayStorage 的目的在于利用分布在网络异构主机之上的不可靠存储资源提供可靠的存储服务。存储资源的不可靠性来源于系统的应用环境,任何提供资源的节点可以随时脱离系统的管理。因此,为了提供安全可靠的存储服务,RayStorage 系统必须充分考虑到以下情况:1)所有存储节点都有可能随时加入或者脱离系统的管理;2)与失效节点之间

收稿日期:2005 - 06 - 16;**修订日期**:2005 - 09 - 13

作者简介:曹楠(1981 -),男,陕西西安人,硕士研究生,主要研究方向:网格计算、计算机网络存储、分布式系统; 康慕宁(1955 -),男,陕西西安人,教授,主要研究方向:软件理论与软件工程.

在本项目开发实践中,我们将这两个作用领域不同的优秀框架进行整合来搭建 J2EE 架构。通过框架的结合,充分发挥了两者的优点,不但使资源得到最大限度的节省和利用,也使得项目开发简洁、结构清晰,并且具备了更好的可扩展性和可维护性。

参考文献:

[1] 夏昕,曹晓钢,唐勇. 深入浅出 HIBERNATE[M]. 北京:电子工业出版社,2005.
[2] 孙卫琴. 精通 HIBERNATE: Java 对象持久化技术详解[M]. 北京:电子工业出版社,2005.
[3] CRAWFORD W, KAPLAN J. J2EE 设计模式[M]. 刘绍华,毛天露,译. 北京:中国电力出版社,2005.
[4] GAMMA E, HELM R, JOHNSON R, *et al*. . 设计模式:可复用面向对象软件的基础[M]. 李英军,马晓星,蔡敏,等译. 北京:机械工业出版社,2000.
[5] TURNER J, BEDELL K. STRUTS Kick Start 中文版[M]. 孙勇,译. 北京:电子工业出版社,2004.
[6] 孙卫琴. 精通 STRUTS:基于 MVC 的 Java Web 设计与开发[M]. 北京:电子工业出版社,2004.

的数据传输必须得到抑制;3)系统所管理节点的失效行为(未通知脱离管理等)具有一定随机性,满足某种统计学规律,属于个体行为,某些节点的失效不应导致整个系统的崩溃。

本文是 RayStorage 系统研究课题的一部分,重点提出了一种全新的主机级的虚拟存储构架 Distributed Virtual Storage Architecture(DVSA),同时定义并分析了系统存储模式。

1　相关研究

在过去的 10 年中,诸如 NAS/SAN[3] 等一系列集中式网络存储系统取代了传统的网络共享存储方式,得到广泛应用,然而传统集中式存的储构架在很好地解决了一系列存储容错问题的同时,仍然无法从根本上解决单边失效所带来的灾难。为了解决上述问题,同时也为了减小存储代价及简化存储管理,诞生了一大批分布式存储系统,早期的系统包括 Bayou[7]、Coda[8] 等,近期的研究成果有 OceanStore[5]、PAST[6] 以及 RDSS[4,11] 等。

在系统构架方面,OceanStore 以及 PAST 构架在文档路由模型[9] 之上,通过分析查找临近节点来实现对主机节点的选取。文档路由模型不需要建立特殊的通讯协议,简化了系统设计,但文献[11]中的研究表明该模型的运行过程大大增加了主机节点计算负载,在应用于海量数据查询及存储的情况下,无法确保运行效率。在 RDSS 中,采用了基于 RAN[10] 的解决方案,采用了传统的 P2P 资源查询方式。应用 RAN 进行资源查询会产生链式反应,波及到大量的其他不相关节点,文献[12]的研究表明,这种方式的查询同样也会增加系统负载,影响存储效率的提高。

在系统可靠性方面,OceanStore、RDSS、CFS[13] 等均选择了维护数据复本的方案来提供可靠的服务。所不同的是,OceanStore 采用了基于 erasure-coding 算法的数据复制方式,通过将数据分成若干数据块,并根据这些数据块计算出冗余码,当数据的任意一部分发生缺失时,能够利用冗余信息及其他数据块重新计算出缺失的部分,这种方案提高了系统的存储资源利用率,但由于算法自身的复杂性增加了系统负载;CFS 采用了基于 master-slaver 的全数据备分模式,对于每一个数据块都维护多份完整的副本,当数据发生变更时,系统在一个原子命令内更新所有的副本,因此 CFS 在保证了系统数据一致性的同时,也产生了较大的响应延迟,降低了使用效率。RDSS 同样采用完整副本备分方案,与 CFS 不同的是 RDSS 制定了一定的策略,仅在必要时才对数据复本进行更新。

RayStorage 在上述两方面做出了很大程度的提高与改进。RayStorage 提出的 DVSA 构架方案在底层采用了基于资源服务器的 P2P 数据通信,避免了资源查询过程中的链式反应,减小了系统响应延迟。同时,为了增强分布式环境下对资源服务器的管理, RayStorage 借鉴了文献[12]的研究成果,以树型结构管理资源服务器,组成服务器的分布式体系,避免了传统 C/S 模式对于服务器的过度依赖。DVSA 构架将系统可靠性的维护提高到一个新的高度,在参考文献[4,5]采用 master-slaver 数据备分方式的基础上,提出了 Logic Volume 间的数据存储模式,建立了数据备分模型,使系统更具可控性,也增强了系统的兼容性与扩展性。

2　RayStorage 简介

2.1　数据模型

RayStorage 系统中使用同一种数据组织形式来实现各个模块间的交互。与文献[4,5]相类似,RayStorage 同样以数据块(block)作为最小数据访问单位,并由若干不同的 block 共同组成一个“逻辑卷”(Logic Volume,LV)。与文献[4]不同的是,RayStorage 中的 LV 不仅仅作为系统的一个存储逻辑单位,更应用于虚拟存储体系之中,以建立更加复杂的数据存储模式。因此,虚拟卷的创建过程有必要先于数据存储过程进行,而文献[4]通过在存储过程中指定 Volume ID 以创建虚拟卷的方式,大大降低了整体可控性。系统在 Manager 节点中分析 LV 信息,并通过 Agent 节点对其中所包含的数据块进行高效的 I/O 操作,提供了安全可靠的 I/O 操作接口,使数据的复制、恢复过程均成为可控的系统机制。

2.2　组织结构

在 RayStorage 系统中,LV 是最基本的逻辑访问及管理单元,拥有全局唯一的卷标号(VID),是存储池中资源的可控子集,并通过存储格式化过程进一步划分出大小可控的 block。存储池是一个容纳了所有 Agent 注册资源索引的全局数据库,记录了当前所有可用存储资源的详细信息,包括资源编号、物理地址、空间大小等,这样的信息元组统称为 SRH(Storage Resource Handler)。block 是系统访问及操作的最小物理单元,它所对应的索引 Block ID(BID)是特定 LV 唯一而非全局唯一的,一个 VID + BID 的组合构成了全局唯一的 block 索引,并通过与 Agent 相关联,建立起逻辑层→物理层的映射。与其他系统不同,为了便于控制,RayStorage 规定每一个 LV 在格式化后其所含 block 的 BID 是固定不变的,而 BID 与 Agent 间的映射根据系统运行状态的不同而动态改变。

RayStorage 系统由 Agent、Manager 以及 Client 三种类型的节点组成:Agent 屏蔽了异构系统间的差异,为 RayStorage 提供单节点存储资源并为上层系统提供高效的 I/O 操作接口;Manager 节点维护 Agent 提供的存储信息,实现资源服务器的功能,建立并管理虚拟卷,定制虚拟存储模式,为用户提供高层次的应用服务;Client 节点实现了对于系统服务的访问及利用,但不为系统中的其他节点提供任何服务。对于网络中的任一主机,既可以扮演单一的角色,也可以同时是上述三种节点的综合。

3　分布式虚拟存储架构 DVSA

RayStorage 提出了分布式虚拟存储构架 DVSA,如图 1 所示。DVSA 作为系统的核心架构,自底向上由物理层、虚拟存储层以及应用层三层构成:物理层为整个系统提供包括存储资源在内的底层基本存储服务,在 RayStorage 中通过建立 Agent 节点来实现 DVSA 物理层的设计,Agent 在管理其空闲共享存储资源的同时,也实现了异构资源的同构化映射;虚拟存储层是 DVSA 的核心,可以进一步细分为存储资源子层、虚拟卷管理子层、存储模式子层,分别用于存储资源信息搜集、分配,管理逻辑卷,定制存储模式,为应用层提供高可靠性的存储服务接口。RayStorage 通过建立存储池及 LV 提供虚拟存储层的数据模型,并在此基础上由 Manager 节点管理这些数据,为用户提供不同的存储模式定制方案,为应用层提供必要的高层存储服务及接口。Manager 节点在系统中扮演了资源服务器的角色,并借鉴了文献[12]的拓扑管理思想,以树型结构管理所有 Manager 节点。所有对于存储管理的请求被发给距离请求节点最近的 Manager 处进行处理,当该 Manager 无法找到请求信息时,系统应用 P2P 多播算法[12] 进行资源查找。当某一 Manager 崩溃时,其他 Manager 节点成为该节点的

后备,确保在分布式环境中系统不会因单边失效而崩溃。

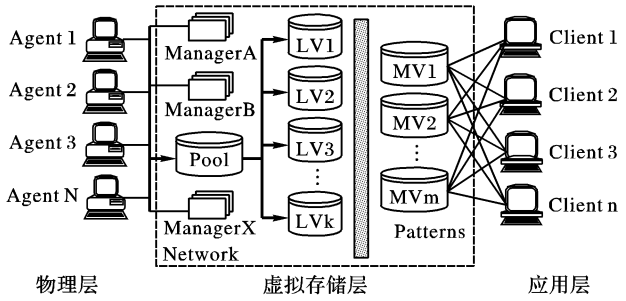

图 1 分布式虚拟存储构架 DVSA

当网络中的一个节点需要加入到系统当中时,该节点必须运行 Agent 节点服务,该服务注册用户指定的存储资源,生成 SRH 信息,并最终由系统统一存放于存储池当中。通过 Agent 注册的资源被称为系统托管资源,资源内部的数据不再受原计算机文件系统的管理,而是直接由 Agent 服务提供必要的资源访问控制及 I/O 服务。在 RayStorage 系统中,Agent 内嵌了一个数据块管理系统,用以实现对托管资源的管理;若 RayStorage 系统中的某个 Agent 节点异常脱离系统,并被资源监测进程检测到时,则 Manager 根据与该 Agent 相关的存储模式,找到其数据副本,执行灾难恢复操作,重新建立起 LV 底层的数据映射关系,使系统恢复到正常的状态。

当系统收到数据读取请求时,Manager 根据用户指定的(VID,BID)查找其对应的 SRH,如果 SRH 有效,则根据 SRH 信息向该节点发送数据读取请求,Agent 服务接收到该请求之后,通过必要的数据完整性及正确性检查后,将其以 P2P 的形式发送到数据请求者(一般为 Client 或者其他 Agent)一端。当收到数据复制请求(replication)时,如果系统位于初始状态,block 与 Agent 间逻辑→物理影射尚未建立,则此时系统通过节点优先级选择算法 NPSA(Node Property Selection Algorithms)计算并选择适当的 Agent,创建 block 与 Agent 间的逻辑→物理影射;否则,根据已知的映射关系直接选定 Agent。在 Agent 确定后,Manager 将相应的 SRH 发送给写入操作请求方,并由其直接建立与 Agent 间的 P2P 连接,进行数据传输通信。

4 存储模式的研究

4.1 存储模式

鉴于底层的分布式结构的不可靠性,为了提供安全可靠的数据存储,DVSA 规定了存储系统中对于数据备分方案的一般性规则、数据备分流程所遵循的策略以及数据间的逻辑拓扑,这些规定被定义为存储模式。

DVSA 采用 Master-Slaver 方式进行数据备分,与文献[4,13]等系统不同的是,DVSA 将 Master 与 Slaver 提升到系统应用的高度,而不仅仅是逻辑上的一个概念。同时,在数据复制方式方面,DVSA 采用的复制策略,既不同于文献[4]的仅在必要时更新的方案,也不同于文献[13]在一个原子操作内完成所有复制过程,系统将复制分为同步与半同步两种级别:同步状态下,当 MV 发生变化时,系统立刻将变化数据反映到 SV 侧;半同步状态下,当 Master 与 Slaver 之间的差分量达到某一阈值时,Manager 才将 Master 中的数据反映到 Slaver 当中,此时进行的是差分拷贝,这种备分方案适合于数据多变的情况,能够迅速快捷的完成数据复制过程。

上述存储策略需要应用在一定的数据模型之上,系统将 LV 根据其担当角色的不同分为 Master Volume、Slaver Volume 及 Raid Volume。两个独立且容量相同的 LV 之间通过建立逻辑上的 Pair 关系形成 MV 及 SV,MV 是用户能够看到并能够使用的逻辑卷,而 SV 用于存放 MV 的副本,一个 MV 可以拥有多个 SV,但一个 SV 有且仅有一个 MV 与之对应,一个 LV 可以既是 SV 又是 MV(此时该 LV 用户不可见,只用于表示其间的逻辑关系),MV 中所包含的数据块 MB 与 SV 中所包含的数据块 SB 通过 BID 一一对应,对于具有相同 BID 的一对 MB 与 SB 来说,与它们具有映射关系的 Agent 所对应的 SRH 互异;而 Raid Volume 是由 Raid 算法构建出的逻辑卷,所涉及到的 LV 之间不再是单纯的 Pair 关系。在这些规定的基础上,DVSA 提出了以下四种存储模式:

1)存储摸式 1

图 2(a)为单副本镜像模式,SV 是 MV 唯一的镜像,每一个数据块具有唯一的一个副本,MV→RV 构成一个 Pair,用于管理 NPSA 值较高的节点,由于这些节点具有良好的稳定性与可靠性,因此无需过多的数据副本。应用这种存储模式,存储空间利用率为 50%,一半空间被用于保存数据副本。

2)存储模式 2

图 2(b)为多复本镜像模式,在 DVSA 中,规定对于一般的计算机节点采用多复本镜像模式,SV 个数从 2 ~ 10 个不等,根据当前 MV 中所包含节点的平均 NPSA 值所决定。在多复本镜像模式下,系统的存储空间利用率为:$\eta = 1/n$,式中 n 代表 SV 的个数。

3)存储模式 3

如图 2(c)所示,串联镜像模式。在系统延迟较小的情况下,应用这种模式能够减少 MV 执行 Replicate 操作时所花费的后台时间,与模式 2 相比而言,MV 只需要执行一次 Replicate 操作便可以由串联效应构成多个数据副本,但模式 3 在系统延迟较大的情况下会产生数据一致性的问题,然而应用这种特性,串联模式可以用于存放数据历史记录。存储模式 3 的存储空间利用率与存储模式 2 相同。

存储模式 1 ~ 3 提供了逻辑镜像备分的 3 种基本模式,它们可以混合构成分布式环境下复杂的镜像存储模式。

4)存储模式 4

Raid[14]冗余模式(如图 2(d)),该模式利用 LV 构筑软件 Raid 进行数据冗余,具有较高的存储资源利用率,适合应用于存储集群内部。对于一般的分布式网络而言,采用这种方案的存储会产生较大的计算开销,降低存储效率。在存储空间利用率方面,存储模式 4 根据 Raid 算法的不同而不同。

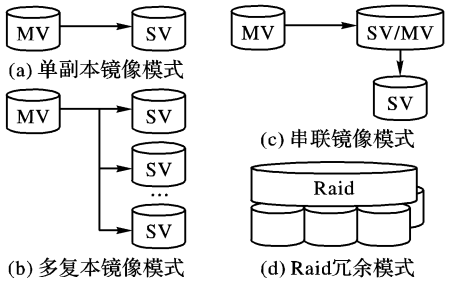

图 2 存储模式

4.2 性能分析

在分布式环境下,系统运行效率往往比存储空间利用率更为重要,因此 RayStorage 系统实现了 DVSA 构架的前三种基本模式。

在 DVSA 的应用中,系统稳定性及存储模式的运行效率

无疑是性能评估的关键,而以往的系统,例如文献[4,5]等,并没有给出系统稳定性方面的评估数据,在对运行效率的评估方面也遵循了不同的评估标准。而在应用 RayStorage 对 DVSA 进行评估的过程当中,系统不失一般性地将稳定性定义为"用户对系统数据的成功访问率":$\mu = k/N$,式中μ代表系统稳定性;k代表用户成功读取系统数据的次数;N代表用户执行读取操作的总数。在该定义下,$1-\mu$体现了在存储于系统当中的用户数据丢失的可能性。同时,系统通过计算用户复制(replicate)或恢复(restore)命令发出后,数据副本更新的总响应延迟来体现存储模式的运行效率。

对 RayStroage 系统根据上述定义进行了一系列仿真试验。在稳定性方面,首先以随机的方式发送数据访问请求,以模仿网络中分散用户对系统数据的随机访问;接着,再以随机的方式中断或开启 Agent 服务,以模拟不稳定的分布式环境;最后,通过系统的长时间运行,统计成功访问次数,并计算出μ,μ 随 SV 个数变更的曲线如图 3 所示(测试环境如表 1 所示)。在存储模式的运行效率方面,系统记载了数据副本响应 replicate 及 restore 请求的时间,根据 SV 个数的不同,运行效率有所差异,如图 4 所示(测试环境如表 2),其中,虚线代表 replicate 操作,实线代表 restore 操作。同时,为了便于与文献[1]进行比较,系统也记录了在整体模式应用下单个 block 更新的响应延迟,对比结果如图 5 所示(测试环境如表 2)。

表 1　系统稳定性实验环境

测试模式	单/多复本镜像模式
LV 容量	0.2GB
SV 个数	1~10 个
网络环境	100M 以太网
Agent 事件频率	每 5s 一次
数据访问频率	每 3s 一次
测试周期	1H
Agent 个数	15 个

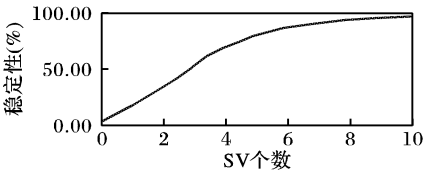

图 3　μ 随 SV 个数变更的曲线

表 2　运行效率实验环境

测试模式	单/多复本镜像模式
LV 容量	0.2GB
SV 个数	1~10 个
网络环境	100M 以太网
读写频率	1 次/min
Agent 个数	15 个
Block Size	4KB/64KB

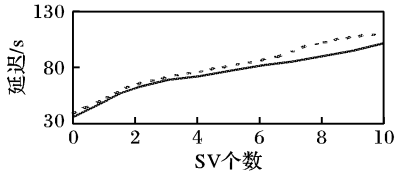

图 4　运行效率随 SV 个数变更的曲线

通过上述实验结果不难发现,在应用 RayStorage 系统时,将 SV 个数设置为 7 时能够确保系统的可靠性达到 90% 以上,而更新响应延迟≤100ms。同时,与 RDSS 的比较发现,RayStorage 在使用大数据块时具有性能上的优势。

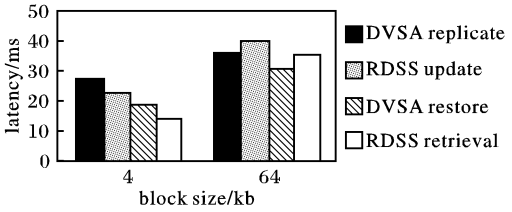

图 5　与文献[1]的对比结果

虚拟存储作为下一代新型存储方式,具有广阔的研究与应用前景,RayStorage 继承了以往这方面的研究经验,今后将在系统构架的不断完善、存储模式的研究与改进及系统相关算法等多方面进行深入地研究。

参考文献:

[1] BOLOSKY WJ, DOUCEUR JR, ELY D, *et al*. Feasibility of a serverless distributed file system deployed on an existing set of destop PCs [A]. Proceedings of Sigmetrics[C]. 2000.

[2] IEEE Storage System Standards Working Group. Virtual Storage Architecture Guide (VSAG)[Z]. 1995.

[3] FARLEY M. Building Storage Networks[M], second edition. Beijing, McGraw-Hill Education Co. and China Machine Press, 2001.

[4] LI XD, LIU C. Towards a Reliable and Efficient Distributed Storage System[A]. IEEE Hawaii International Conference on System Sciences, 2005.

[5] KUBIATOWICZ J, BINDEL D, CHEN Y, *et al*. OceanStore: An Architecture for Global-Scale Persistent Storage[A]. In Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems(Cambridge, MA)[C]. ACM Press, 2000.

[6] ROWSTRON A, DRUSCHEL P. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility[A]. Proceedings of ACM SOSP[C]. 2001.

[7] DEMERS A, PETERSEN K, SPREITZER M, *et al*. The Bayou architecture: Support for data sharing among mobile users[A]. Proceedings of IEEE Workshop on Mobile Computing Systems and Applications[C]. Santa Cruz, California, 1995.

[8] KISTLER JJ, SATYANARAY M. Disconnected operation in the Coda file system[J]. Thirteenth ACM Transactions on Computer Systems, 1992, 10(1):3-25.

[9] PLAXTON CG, RAJARAMAN R, RIHCHA AW. Accessing Nearby Copies of Replicated Objects in a Distributed Environment[R]. Technical Report: CS-TR-97-11, University of Texas at Austin, Austin, TX, USA, 1997.

[10] LIU C, WELCH L, JUEDES D. The Resource Area Network Architecture Pattern[A]. the 10th Conference on Pattern Languages of Programs[C]. Urbana, IL, USA, September, 2003.

[11] LI XD, LIU C. RDSS - A reliable and efficient distributed storage system[D]. Master's degree, Computer Science and Technology department of Ohio University, 2004.

[12] TU WQ, JIA WJ. A Scalable and Efficient End Host Multicast Protocol for Peer-to-Peer Systems-DSCT[A]. Proceedings of IEEE Global Telecommunications Conference, GLOBECOM '04[C]. 2004.

[13] DABEK F, KAASHOEK MF, KARGER D, *et al*. Wide-area cooperative storage with CFS[A]. Proceedings of ACM SOSP[C]. 2001.

[14] SHINKAI Y, MARUYAMA T. Software RAID Technology for cluster environments[A]. 18th IEEE Symposium on Mass Storage Systems and Technologies(MSS'01)[C]. 2001.