一种基于存储虚拟化的异步远程镜像系统

向小佳 余宏亮

(清华大学计算机科学与技术系 北京 100084)

摘 要 信息数据在当今社会中的重要性日益提高.远程镜像系统通过配置冗余的硬件和相应的高可靠软件,能够保证信息系统的数据安全.但是,已有的远程镜像系统都部分存在着依赖于专用设备或底层驱动、性能差等不足.该文提出一种应用于存储区域网络环境的、基于存储虚拟化的异步远程镜像系统.首先,设计了异步镜像逻辑卷,作为虚拟化的数据容器,该卷能够自动追踪并异步传输数据更新,节省网络带宽,同时不依赖于任何的底层设备和驱动.其次,提出了一种不间断服务的异步镜像协议,支持服务在镜像主节点和从节点间的无缝迁移,支持灾难发生和灾后恢复时主从节点的自动切换和不间断服务,同时,协议完全在镜像节点内执行,不影响客户主机的性能.最后,针对镜像卷的读写特点,设计了基于锁的镜像卷访问控制机制,既保证数据一致性,又能在此前提下尽可能提高读写并发性.FTP trace(文件传输服务跟踪记录)的重放实验证明,在人为引入主节点失效的情况下,服务能够无缝迁移;在主节点灾后数据恢复进行的同时,能够保证服务不间断;最后,FTP Trace 的重放流程在"主节点失效-服务迁移-灾后恢复"场景下仅比无故障时多花费 14%的时间.

关键词 异步镜像逻辑卷;增量卷;主/从节点

中图法分类号 TP302 **DOI 号**: 10.3724/SP.J.1016.2009.01905

An Asynchronous Remote Mirror System Based on Storage Virtualization

XIANG Xiao-Jia YU Hong-Liang

(*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084)

Abstract Computerized data becomes more and more critical in today's digital life. Remote mirror can keep information systems alive with redundant hardware and software, even encountering site corruptions. Unfortunately, existing mirror systems have some limitations such as relying on specific storage devices, poor performance, and so on. This paper presents a storage virtualization based asynchronous remote mirror system. First, in this system, all data are protected by a logical container, that is, an asynchronous mirror logical volume, which can trace and transfer data updating automatically. This way, there is no device and driver dependency at all. Secondly, this paper presents an advanced non-stop mirror protocol to support seamless service switching when site corruption happens and automatic data recovery at rebuilding process after disaster. In addition, all the protocol can be executed on mirror IO nodes in SAN, introducing little overhead to the protected servers. Finally, it gives design on lock based concurrent access control mechanism for the asynchronous mirror logical volume. With the help of the lock, not only data consistency can be maintained, but also good concurrent read/write performance can be achieved. Experiments and evaluation results demonstrate that the prototype system has the ability to provide high reliability while only introduce little overhead: tolerated a site corruption while kept providing service, costing only 14% more time.

Keywords asynchronous mirror logical volume; increment logical volume; primary/secondary sites

收稿日期:2009-07-15;最终修改稿收到日期:2009-08-26.本课题得到国家"九七三"重点基础研究发展规划项目基金(2007CB311100)资助.**向小佳**,男,1977 年生,博士研究生,主要研究方向为存储网络、分布式文件系统、容灾恢复以及虚拟化技术等.E-mail: xiangxj05@mails.tsinghua.edu.cn. **余宏亮**,男,1976 年生,博士,副教授,主要研究方向为分布式系统、容灾恢复等.

1 引 言

在现代生活中,海量的信息数据变得越来越重要,甚至关乎企业的存亡.用来保护数据,提高系统可靠性的各种技术也越来越受到关注,这些技术包括备份、快照、连续数据保护 CDP(Continuous Data Protection)、远程镜像等等.

在这些技术中,备份系统通过定期在备份节点上保存数据副本来提高系统的可靠性,代表系统包括 Veritas Volume Replicator[1]、IBM 公司的 Peer to Peer Remote Copy① 等.快照系统为轻量级的数据保护技术,通过 COW(Copy On Write)和 ROW(Redirect On Write)的方式来生成数据的历史映像,但是,当遭遇较严重数据灾难时,如存储介质损毁,快照技术必须和其他数据保护技术(如备份)相结合才能完整恢复被保护节点的数据.快照系统的代表有 Clotho[2]、EMC 公司的 Snapview② 以及微软公司的卷影拷贝系统 Volume Shadow Copy③ 等.CDP 技术通过日志等方法保留所有的数据更改记录或数据的历史版本,为系统提供恢复到任何时间点的能力.但是,CDP 系统需要记录所有历史数据,不仅空间消耗巨大,对历史数据的访问也存在性能瓶颈.而且,在严重数据灾难如存储介质损毁、节点失效时,CDP 技术同样需要与备份等技术结合才能有效完整地恢复节点数据.CDP 系统的代表包括 Sweeper[3]、IBM 公司的 Tivoli④ 等.远程镜像系统通过在异地保留数据的一致副本,能过容忍节点失效等数据灾难,同时,通过数据的重定向,支持主节点失效时服务向从节点迁移.

当前已存在的镜像系统都存在如下的部分缺点.首先,有的镜像系统依赖于底层的设备或驱动.例如:EMC 公司的 SRDF[4] 镜像系统必须构建于本公司的 Symmetrix 网络存储系统之上;SnapMirror[5] 是一个基于 COW 快照技术的镜像系统,能够利用快照技术快速生成数据映像,并以此映像为镜像数据源进行异步数据拷贝,性能好.但是该系统依赖于 NetAPP 公司的存储设备以及嵌入在设备底层的 WAFL[6] 文件系统.其次,镜像系统的镜像数据流会与被保护应用的正常读写数据流争抢资源,影响性能,尤其是同步镜像数据流会在一定程度上影响系统的写性能.例如:LVM⑤ 系统中的镜像模块就采用了同步镜像协议,而且该模块会消耗客户主机的资源进行地址映射、镜像元数据的更新以及镜像状态的维护.针对性能上的缺点,Seneca[7] 镜像协议采用了异步数据传输机制,并提出了写合并(write coalescing)技术,本文也借鉴了这些思想.

我们设计了一种应用于存放海量数据的存储区域网络环境、基于存储虚拟化技术的远程异步镜像系统;其中设计了一种位于虚拟块设备层的异步镜像逻辑卷,通过在底层的异构存储资源池中构建该卷,向上层提供标准块设备接口,不但管理灵活,同时没有对设备和驱动的依赖性;异步镜像逻辑卷内部集成了更小的逻辑数据容器——增量卷,能够追踪记录并合并一段时间窗口内的数据更新,提供异步数据镜像时的数据源,既有效支持了在线服务,又节省了网络带宽资源.我们还提出了一种不间断服务的异步镜像协议,支持服务在镜像主从节点间的无缝迁移,支持灾难发生和灾后恢复时主从节点的自动切换和不间断服务,同时,协议完全在镜像节点内执行,不影响客户主机的性能.最后,针对异步镜像逻辑卷的读写特点,设计了基于锁的镜像卷访问控制机制,保证多主机并发访问时数据的一致性.

本文第 2 节介绍系统的体系结构;第 3 节详细论述异步镜像逻辑卷、不间断服务异步镜像协议以及基于锁的镜像卷访问控制机制 3 个关键技术;实验与结果在第 4 节介绍,最后是结论.

2 体系结构

我们的镜像系统能够部署在基于存储区域网络的集群环境中,保护大型数据中心的共享存储资源池.其体系结构如图 1.

图 1 中,WAN(Wide Area Network)将整个网络分成本地和远端两个区域.在本地,客户主机是存储区域网络的使用前端,可以是普通的 PC(Personal Computer)或工作站,集成有逻辑卷管理模块,能够与虚拟化块设备进行正常交互.镜像主节点是存储区域网络中的 IO(Input/Output)节点,将由其管理的存储资源池(包含各种异构存储设备)通过光纤或以太网数据链路开放给客户主机使用.在客户主

① PPRC Migration Manager Services. http://www-03.ibm.com/systems/storage/solutions/services/featured/pprc_mm.html
② Snapview. http://www.emc.com/products/detail/software/snapview.htm
③ Volume Shadow Copy Service. http://msdn.microsoft.com/en-us/library/aa384649(VS.85).aspx
④ Tivoli. http://www-01.ibm.com/software/tivoli
⑤ LVM2. http://sources.redhat.com/lvm2

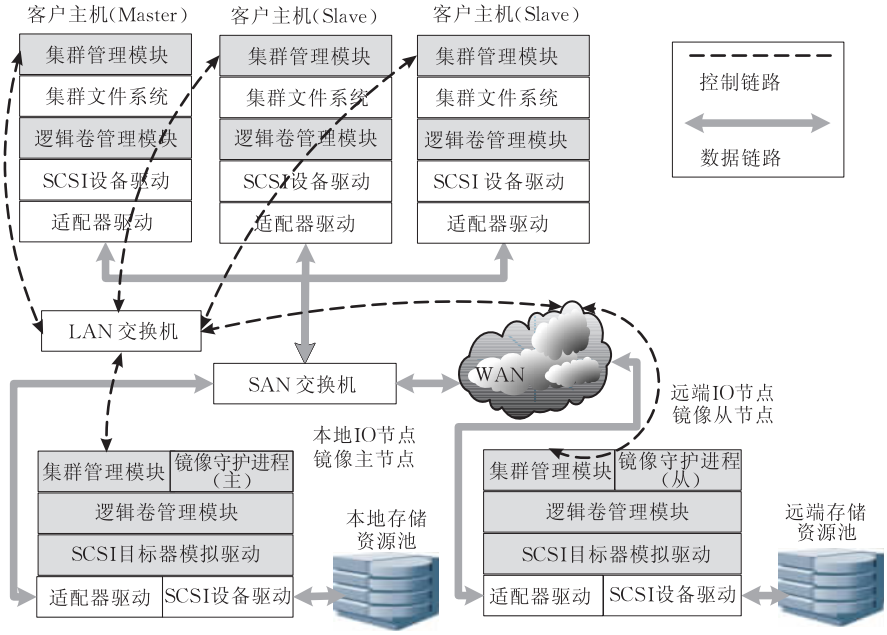


图 1　系统体系结构

机的视图中,镜像主节点提供的海量数据存储服务以标准块设备接口的形式呈现,这些块设备是异步镜像逻辑卷,但其实现的物理细节对客户主机透明.在正常情况下,镜像主节点就是本地 IO 节点,其存储资源池也在本地.在远端,远端 IO 节点负责管理远端的存储资源池,正常情况下作为镜像从节点维护与主节点各异步镜像逻辑卷一致的数据副本.从节点也具备将自己维护的卷开放给客户主机使用的能力,但在正常情况下,这些卷是对客户主机透明的.

图 1 中,所有节点通过控制链路构成一个集群,客户主机节点拥有对异步镜像逻辑卷并发读写的能力.我们的实现中,读写访问的控制是通过锁机制来完成的.集群中各节点通过投票选择算法,能够在所有客户主机节点中选出一个主节点 Master 和若干从节点 Slave,共同进行锁的管理.同时,集群是高可靠的,节点发生故障时集群拓扑会动态变化,屏蔽故障节点,选出新的 Master 或 Slave,且保证集群中拥有大多数的存活节点,避免由于通信链路引起集群分割(partition).Master 节点的主要功能是:(1)发起周期性的心跳探测,维护集群结构;(2)维护锁管理元数据,处理其它节点发过来的锁请求.Slave 节点的主要作用是维护锁管理元数据的完全一致的副本,作为候选锁管理者在 Master 故障时能够自动接管锁的管理,维持集群的正常运作.

图 1 中,本地和远端 IO 节点分别担任镜像的主节点和从节点,通过跨越 WAN 的数据链路传递镜像数据.主从节点在故障发生时会互相切换,其典型流程可以概括为“主节点失效-服务迁移-灾后恢复”,具体描述如下:首先,正常工作状态下,镜像系统的本地和远端 IO 节点都正常运作,此时,本地 IO 节点也作为镜像主节点向客户主机提供存储服务,记为“主/本地”节点;远端 IO 节点作为镜像从节点,记为“从/远端”节点.接下来,假设本地节点损毁,此时远端节点会升级为主节点,向客户主机提供存储服务,记为“主/远端”节点,而系统中暂时缺少从节点.最后,本地节点的相关设备被修复,加入到系统中,记为“从/本地”节点;此时,在“主/远端”节点向客户主机提供存储服务的同时,“从/本地”节点也通过“主/远端”节点进行数据恢复.当数据恢复完成,主从节点再次切换,“主/远端”、“从/本地”切换为“主/本地”和“从/远端”节点,此时系统恢复正常工作状态.

图 1 中,灰色的模块为我们开发或修改的模块,其它模块为操作系统或应用系统自己的功能模块.在客户主机端,集群管理模块工作于用户态,主要负责维护本机在集群中的状态,接收、处理和发送控制消息.逻辑卷管理模块工作于内核态,调用下层标准 SCSI(Small Computer System Interface)设备驱动访问存储区域网络中的共享资源,负责发起异步镜像逻辑卷的创建和维护命令,也负责向上层应用提供标准块设备接口.

在镜像主节点中,主要的模块是:集群管理模块、镜像守护进程、逻辑卷管理模块、SCSI 目标器模

拟驱动.集群管理模块的功能同客户主机端.镜像守护进程工作于用户态,负责监控本节点的工作状态,按照不间断服务异步镜像协议(见 3.2 节)控制节点的状态转移,同时协调数据在镜像主从节点间的传递.逻辑卷管理模块的功能与客户主机端的相应模块类似,不同之处在于,位于 IO 节点的逻辑卷管理模块是异步镜像逻辑卷的实际管理模块,具有逻辑卷的内部结构知识,是卷间镜像数据流控制、卷元数据管理的实际执行者,而异步镜像逻辑卷的内部细节对客户主机端的逻辑卷管理模块透明.SCSI 目标器模拟驱动主要负责管理存储区域网络中的异构存储资源池,接收来自各个节点的块请求,将资源池的存储资源模拟成 SCSI 存储设备来响应这些块请求.

镜像从节点的软件框架类同于主节点,也包含这 5 个模块,其不同之处在于:(1)从节点的镜像守护进程在系统正常运作时的主要工作是监视系统状态;一旦"主/本地"节点失效,其立即触发服务迁移,接管客户主机的 IO 请求;另外,数据恢复操作也是由从节点的镜像守护进程负责触发的.(2)从节点中的SCSI目标器模拟驱动在正常情况下只负责响应来自"主/本地"节点的块请求,屏蔽来自客户主机的请求;但主节点发生故障后,该模块必须立即解除屏蔽,以提供存储服务.

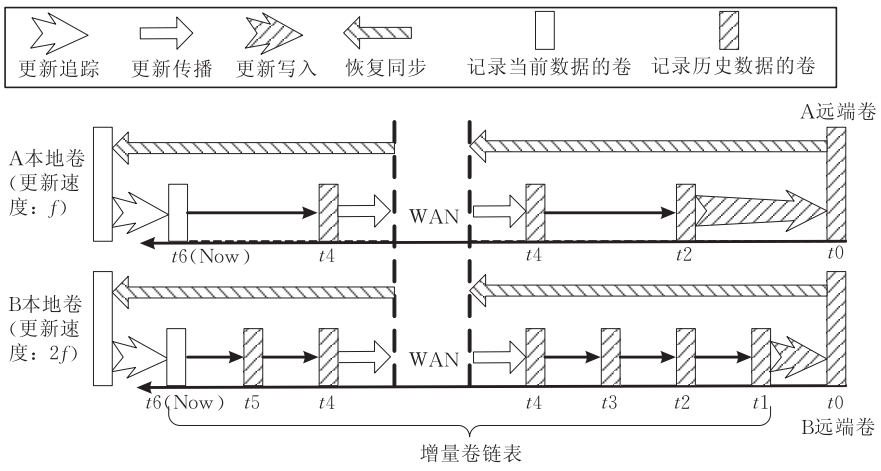
3 关键技术

首先,我们详细介绍异步镜像逻辑卷的内部结构以及基于其结构的镜像数据流.其次,给出了不间断服务异步镜像协议的状态机.最后,描述了基于集中式锁的镜像卷访问控制机制.

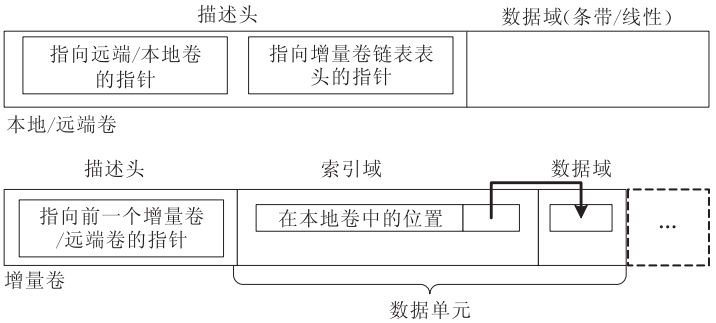
3.1 异步镜像逻辑卷

异步镜像逻辑卷是数据的基本容器,给客户主机提供标准块设备的接口,为存放在其中的数据生成一致的镜像副本.

异步镜像逻辑卷也是一种复合逻辑卷,由其它逻辑卷构成.图 2(a)示例了两个异步镜像逻辑卷 A 和 B 的构成,它们主要包含如下部分:(1)本地卷.该卷位于本地 IO 节点,通过在本地存储资源池中划分资源而生成;正常工作情况下,本地 IO 节点也就是镜像主节点,本地卷向客户主机提供存储服务;(2)远端卷.该卷位于远端 IO 节点,在正常工作情



(a) 镜像逻辑卷构成框架



(b) 卷内部结构

图 2 异步镜像逻辑卷结构

况下,该卷仅为本地卷提供镜像服务,仅当主节点失效导致本地卷无法工作时,远端卷才会开放给客户主机使用;(3)增量卷.该卷主要用来记录和合并数据更新,同时作为镜像数据异步传输的承载容器.所有增量卷都具有时间标签,用来标志其所记录的数据更新的时间窗口,例如,图 2(a)中,时间标签为 $t1$ 的增量卷记录了时间窗口为$(t0,t1]$这个区间内的镜像数据更新.所有增量卷按时间排序,构成增量卷链表.链表的表头为当前增量卷,具有最新的时间戳,正在记录数据更新;其它增量卷皆为历史卷,记录了过去一段时间内的数据更新,会依序被传输到远端并合并到远端卷内.

如图 2(b),本地卷和远端卷的内部结构与普通逻辑卷相似,包含描述头和数据域两部分.数据域中逻辑地址与物理地址的映射关系既可以是条带的,也可以是线性的.描述头中除包含卷的容量、映射方式等属性外,主要包含指向对方,即远端/本地卷的指针,对于本地卷,还应包含指向增量卷链表表头的指针.

增量卷的内部结构包含一个描述头和若干数据单元.描述头记载了该增量卷的元数据,除了用来控制和管理增量卷空间占用以及逻辑地址和物理地址之间映射关系的元数据,主要还包括时间戳和指向前一个增量卷或远端卷的指针.数据单元用来装载需要传递的镜像数据更新.一个增量卷可以包含若干个数据单元,数据单元的数量随着更新数据量的增加而增加.每个数据单元包含两个域:索引域和数据域.数据域存储的是更新后的数据块;索引域包含两个指针,记录了数据的映射关系,一个指针指向数据域中的一个数据块,另一个指针用来记录该数据块在本地卷中的位置.设置索引域集中存放数据映射关系,便于对索引关系的检索与管理:例如,当有新的数据块更新时,我们就可以快速搜索索引域以查询该数据块是否已被更新过,若是,直接用新的内容覆盖,若不是,则在数据域分配新的数据块并在索引域建立映射关系.

异步镜像逻辑卷支持的数据操作包括:更新追踪、更新传播、更新写入与恢复同步.如图 2(a)所示,更新追踪指的是"主/本地"节点在提供数据存储服务的同时,实时地将数据的更新记录在增量卷链表的表头增量卷中;更新传播指的是数据更新周期性地从"主/本地"节点传输到"从/远端"节点;更新写入指的是"从/远端"节点将接收到的增量卷中的数据更新写入到远端卷中;以上 3 个操作是在系统正常工作状态下并发执行的.恢复同步指的是灾后恢复过程中将远端卷中的数据整体恢复到刚修复的本地卷的过程;在我们的实现中,基于哈希的差异拷贝技术被用来减少恢复同步时数据的传输量,节省网络带宽:对于本地卷中数据部分损毁的情况,按照协议先传输待恢复数据块的粗粒度 Hash 值,若本地卷中的待恢复块的粗粒度 Hash 值与之不同,则恢复相应数据块;否则传输待恢复数据块的精确 Hash 值,做第二次匹配,若不匹配,则恢复数据块,否则,说明该数据块未被损坏,无需恢复.上述所有 4 种操作都是在镜像主从节点上进行的,对客户端完全透明,不影响其性能.

增量卷的生成频率是一个需要动态调节的参数,它影响到两个指标:(1)镜像数据传输的网络带宽占用率;(2)"从/远端"节点所维护的远端卷与本地卷间的一致性.如果某异步镜像逻辑卷的增量卷生成频率越高,系统对其调度也越频繁,必然会更多地占用网络带宽,但优点是远端卷更新快,与本地卷间的一致性好;反之,则节省带宽,但是远端卷与本地卷间的数据差异较大.

异步镜像逻辑卷中增量卷生成频率调节的基本原则是:(1)使用增量卷容量阈值来决定新增量卷生成的时间点.由于增量卷记录的是累计数据更新,则其容量反映了远端卷和本地卷间的数据差异,当其容量超过阈值时,说明累积的数据差异已经足够大,系统停止其上的更新追踪,准备更新传播,同时触发生成新的增量卷.这样能够在不同的更新传播任务间合理地分配带宽,如图 2(a),异步镜像逻辑卷 B 的更新速度是 A 的 2 倍,按照基于容量阈值的方法,B 的增量卷生成频率是 A 的 2 倍,则系统会为其分配更多的带宽用来传输数据更新,进而使 A 和 B 的远端卷和本地卷间的数据差异维持在相当的水平.(2)通过动态调节容量阈值来寻求网络带宽占用率和远端/本地卷一致性间的平衡.镜像守护进程监控增量卷链表的长度,如果过长,则调高容量阈值,使增量卷能够记录和吸收较长时间窗口内的数据更新,降低增量卷的生成频率,以增加远端卷和本地卷间数据差异为代价来缓解系统带宽的压力;相反,则降低容量阈值,提高增量卷生成和更新传播的频率,通过使用更多的带宽来保证远端卷和本地卷间更高的一致性.

3.2　不间断服务异步镜像协议

本部分首先列出与该协议相关的状态和事件的集合,接下来,我们通过"主/本地"节点失效引发镜

像主从节点切换,再到灾后恢复的场景,详细阐述协议状态机的状态转移流程.

图 3 所示为我们的不间断服务镜像协议状态机.状态机中的状态列表见表 1,事件列表见表 2.状态转移如图 3 中的箭头所示.其中,实线箭头主要代表系统的正常工作流程,另外三种箭头需要注意:(1)带虚线的箭头表示的状态转移最终会导致镜像系统无法容忍的灾难,即主从节点皆失效的 DISASTER 状态,对该状态的处理不在本文论述的范围内.(2)加粗的黑实线箭头代表在上述"主节点失效-服务迁移-灾后恢复"场景中本地节点的状态转移流程.(3)加粗的灰实线箭头代表在该场景中远端节点的状态转移流程.

如图 3,本地节点在"主节点失效-服务迁移-灾后恢复"场景下的状态转移流程描述如下:(1)系统工作正常,"主/本地"节点状态为 NORMAL_PRI.(2)Prefailed 事件发生,主节点故障,其工作状态转移为 NONE,角色切换为从节点,等待修复;而原来的从节点则接替主节点继续提供服务;该状态下镜像系统中存在"主/远端"和"从/本地"节点,取代了正常状态下的"主/本地"和"从/远端"节点.(3)当事件 Secrepaired 发生后,"从/本地"节点设备修复

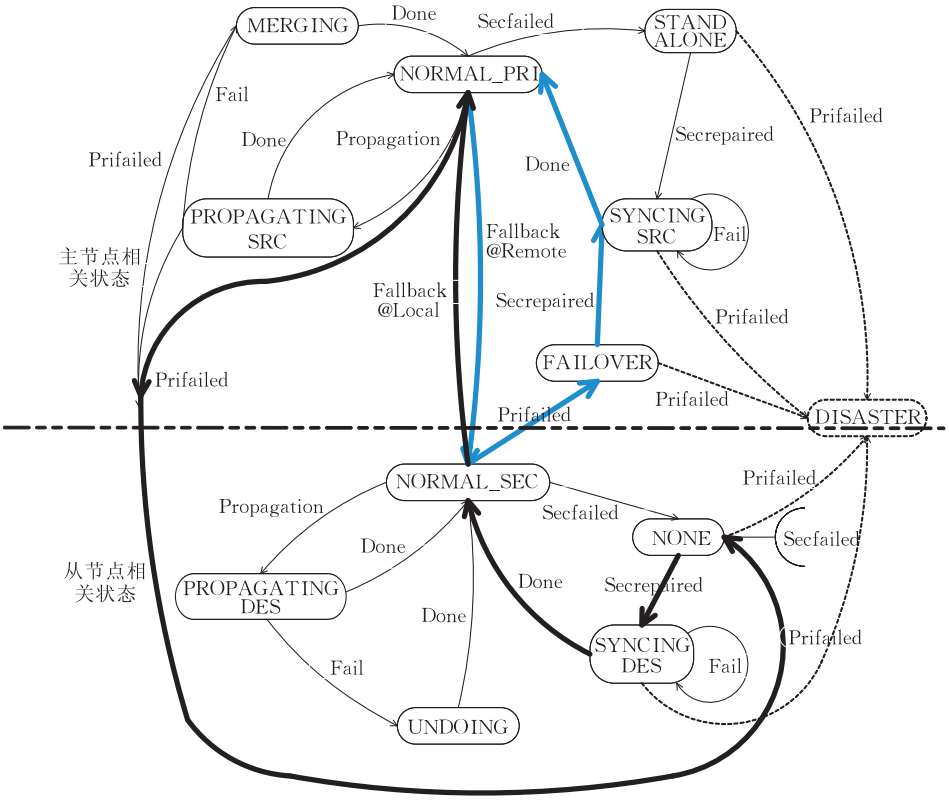


图 3 不间断服务镜像协议状态机

表 1 状态列表

	状态	描述
主节点相关状态	NORMAL_PRI	主节点的正常工作状态(无故障发生)
	PROPAGTING_SRC	主节点在更新传播中作为数据更新发送方的工作状态
	MERGING	当更新传播失败时,将未传播的数据更新与新产生的待传播更新合并,以备下一次更新传播操作使用
	STANDALONE	当从节点失效时,主节点独立工作
	SYNCING_SRC	在恢复同步过程中,主节点作为源,向从节点提供恢复数据的工作状态
	FAILOVER	灾难发生且主节点故障时,原来的从节点通过服务切换成为新的主节点,继续提供服务的工作状态
从节点相关状态	NORMAL_SEC	从节点的正常工作状态(无故障发生)
	PROPAGTING_DES	从节点在更新传播中作为数据更新接收方的工作状态
	UNDOING	从节点在更新传播失败后,回滚记录,等待下一次更新传播
	NONE	空闲状态
	SYNCING_DES	在恢复同步过程中,从节点作为目的,向主节点请求恢复数据的状态
	DISASTER	主从节点皆失效

表 2　事件列表

事件	描述
Done	完成事件
Fail	失败事件
Propagation	发起更新传播
Prifailed	主节点失效
Secfailed	从节点失效
Secrepaired	从节点设备被修复并投入使用
Fallback@Local	恢复同步过程结束后,当本地节点发现自己仍然处于从节点的地位,则发起切换请求,要求转变为主节点,恢复镜像系统的正常态
Fallback@Remote	在恢复同步过程结束后,“主/远端”节点响应本地节点的切换请求,转换为“从/远端”节点

完成,开始数据恢复,其状态转移为 SYNCING_DES,开始向主节点请求恢复数据,恢复同步操作开始.(4)当恢复同步成功结束,“从/本地”节点转移为 NORMAL_SEC 状态,具备服务提供能力,此时主从节点又同时在线了.(5)“从/本地”节点检测到自己的从节点状态,触发 Fallback@Local 事件,向主节点发起切换请求,当主节点正确响应后,本地节点转移为 NORMAL_PRI 状态,完成“从/本地”到“主/本地”节点的切换.

如图 3,远端节点在“主节点失效-服务迁移-灾后恢复”场景下的状态转移流程描述如下:(1)系统正常工作,“从/远端”节点状态为 NORMAL_SEC.(2)Prefailed 事件发生,主节点故障,远端节点工作状态转移为 FAILOVER,向客户主机提供数据服务,由“从/远端”变为“主/远端”节点.(3)当事件 Secrepaired 发生后,恢复同步开始,远端节点作为数据源,其状态转移为 SYNCING_SRC.(4)当恢复同步成功结束,主从节点又同时在线了,远端节点状态转移为 NORMAL_PRI.(5)Fallback@Local 事件发生,“主/远端”节点响应“从/本地”节点的请求,开始角色切换,状态转移为 NORMAL_SEC,切换为“从/远端”节点.

3.3　镜像卷的访问控制机制

在存储区域网络中,异步镜像逻辑卷可以为若干客户主机所共享.如图 1,由于异步镜像逻辑卷被应用于存在并发访问的集群环境,访问控制机制与存储系统的一致性息息相关.在我们的实现中,采用了集中式的高可靠锁机制.同时,除了支持读写访问外,异步镜像逻辑卷还支持如下两类操作:更新传播与恢复同步,所以其访问控制机制需要进一步扩充.

在如图 1 的客户主机集群中,锁的管理权限赋予给 Master 节点,而两个 Slave 节点作为候选锁管理者都保留有锁管理元数据的完全一致的副本.锁管理元数据主要包含一个锁记录表,其中的每条记录与一个存储资源单位对应,如共享卷中存放的某个文件.每个锁申请都会生成一个锁记录,以申请访问的存储资源在地址空间中的唯一标识为键值,通过 Hash 的方式插入到锁记录表中.

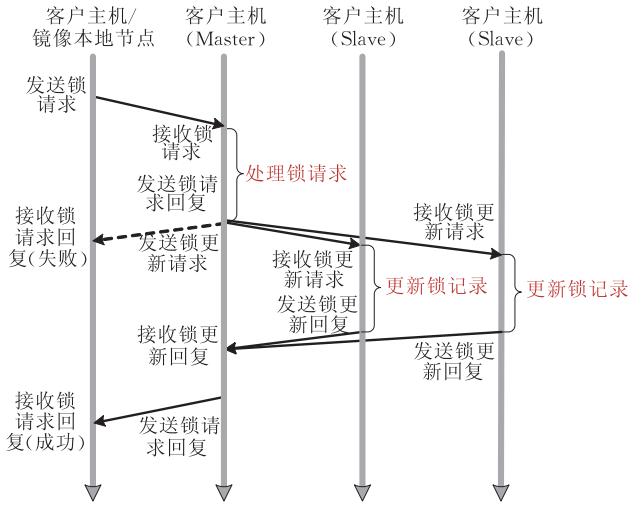


图 4　锁申请流程

锁的管理是集中式的,图 4 所示锁申请流程如下:(1)发起端需要访问共享存储资源,例如,客户主机需要读写异步镜像卷中的数据,必须先向 Master 发送锁请求.(2)Master 接收到锁请求后,通过查询锁记录表,检查对锁的申请与已有记录的冲突情况,最后决定是否授予请求方相应的访问权限,如果产生冲突,则直接回复请求失败,否则进入后续流程.(3)Master 在锁记录表中添加新的记录,代表申请者对存储资源拥有访问权,同时将命令所有 Slave 节点同步更新其锁记录表.(4)Slave 节点接收到锁更新命令后,执行相应的元数据修改,并回复 Master 锁更新成功.(5)Master 确认所有 Slave 更新完毕,则向发起端回复锁请求成功,至此,发起端拥有了存储资源的相应访问权限.上述流程中的第 4 步,Slave 也会出现故障,不响应 Master 的锁更新命令,从而导致锁申请的失败.该过程涉及集群拓扑的变化,需要根据相应的选举算法选出新的 Slave 节点,不在本文的论述范围内.

表 3 锁兼容矩阵

	读锁	写锁	增量卷锁	恢复锁
读锁	1	0	1	1
写锁	0	0	0	0
增量卷锁	1	0	—	—
恢复锁	1	0	—	—

Master 对锁的处理主要是检查待申请锁与已持有锁之间的兼容性,其关键数据结构为锁的兼容矩阵.为支持镜像卷的特有操作,除了读写锁以外,兼容矩阵中扩充了两类锁:(1)增量卷锁,用来支持更新追踪操作.其应用场景如下,每当新的增量卷生成时,新的增量卷会取代原来的增量卷链表表头,导致数据更新的写入位置发生变化,从而与写操作发生冲突,"主/本地"节点通过向 Master 申请增量卷锁,能够控制写操作与增量卷生成的顺序执行,保证数据的一致性.(2)恢复锁,用来支持恢复同步操作.其应用场景如下,当"从/本地"节点灾后修复完成,向"主/远端"节点请求恢复数据,对镜像远端卷的读取与正在该卷上执行的写操作发生冲突,"从/本地"节点通过申请恢复锁,能够暂时阻塞镜像远端卷上的后续写操作,保证恢复同步结束后"从/本地"节点能够顺利切换为"主/本地"节点并提供服务.

锁兼容矩阵中,'1'说明两类锁是兼容的,相应操作可以并发执行;'0'代表两类锁是冲突的,相应操作只能按发起时间顺序依次执行.如表 3:(1)除了写锁外,读锁与自身以及扩充的两类锁都兼容.(2)写锁与任何锁都是冲突的.(3)增量卷锁与读锁兼容,与写锁冲突,同时需要说明的是:首先,由于同一异步镜像逻辑卷中,增量卷在某一时刻只可能生成一个,所以增量卷锁间的兼容性无定义;其次,增量卷锁适用于正常工作状态下,而恢复锁适用于容灾恢复过程中,两种锁不会存在于同一场景下,故其兼容性无定义.(4)恢复锁仅与读锁兼容,而其与增量卷锁和自身的兼容性无定义.

4 实验与评测

我们的实验用原型系统实现细节如下:逻辑卷管理模块、目标器模拟驱动皆工作于内核态,以可加载模块的形式存在,其中,逻辑卷管理模块的向上接口封装为系统调用形式.镜像守护进程用 perl 脚本来构建,仅针对实验实现了对更新传播操作、恢复同步操作和"主节点失效-服务迁移-灾后恢复"场景的支持.

实验环境包含一个客户主机和两个 IO 节点,其中一个 IO 节点为本地节点,另一个为远端节点.IO 节点管理的存储资源通过 iSCSI(Internet Small Computer Systems Interface)协议开放给客户主机使用.主机与 IO 节点间用千兆以太网相连.为模拟长度为 100 公里的镜像数据线路,我们给本地和远端节点间传送的每一个 IO 请求都加上了一个 2ms 的延时.实验用客户主机的配置为:一个 Xeon 2GHz 的 CPU,512MB 内存,一块 34GB 的 Seagate SCSI (ST336607LW)硬盘.两台 IO 节点的配置相同:一个 Xeon 2.33GHz 的 CPU,16GB 内存,1TB×6的 Seagate SATA 硬盘.操作系统都使用 Linux,内核版本为 2.6.18.

实验方法如下:

首先,在客户主机,我们使用 Bonnie++[①]来测试异步镜像逻辑卷的读写性能,以比较该卷同普通的存储区域网共享网络磁盘之间的性能差异.

然后,我们在客户主机上安装一个 FTP 服务器,在存储区域网络中创建一个异步镜像逻辑卷,用来保护 FTP 服务器的存储资源库(160GB),即,令 FTP 的存储资源库位于本地卷,资源库的镜像副本位于远端卷.我们通过重放 FTP Trace 的方法来检测镜像系统的容灾功能以及其服务性能.实验所使用的 FTP Trace 文件取自清华大学计算机系学生 FTP 站点的真实负载记录,我们截选了其中的一段,共包含 55788 个数据传输会话,涉及到的数据总量达 40GB.我们在两个场景下重放该 Trace 文件:(1)FTP 存储资源库部署于普通网络磁盘之上,且重放过程中无事故发生.(2)FTP 存储资源库部署于异步镜像逻辑卷内,且重放过程中出现了一次主节点失效,即整个系统经历了一次"主节点失效-服务迁移-灾后恢复"的全过程.我们以重放时间作为两种场景下系统性能的一个度量,这样,不但能够验证镜像系统对数据和服务的保护能力,还能够得出故障发生时整个系统的服务能力下降程度.

4.1 BenchMark 测试

本测试中,我们使用了 Bonnie++的 5 个微测试.(1)按字符顺序写.以字符为单位顺序写一个新文件,调用标准库 libc 中的 putc();(2)按块顺序写.以块为单位顺序写一个新文件,调用标准库 libc 中的 write();(3)重写.顺序读写文件的每一块,该文件为已存在文件,操作顺序为先读后写,调用标准库 libc 中的 read()和 write();(4)按字符顺序读.以字

① Coker R. Bonnie++. http://www.coker.com.au/bonnie++

符为单位顺序读文件,调用标准库 libc 中的 getc();(5)按块顺序读.以块为单位顺序读文件,调用标准库 libc 中的 read().在客户主机上,这些测试分别在挂载了网络磁盘和异步镜像逻辑卷的不同目录下执行,运行结果给出了各测试的平均 IO 吞吐率.

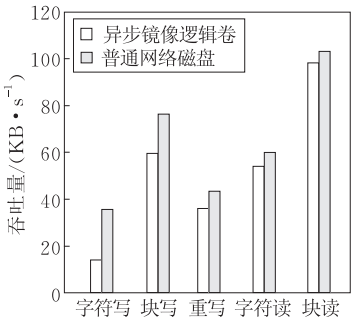


图 5　Bonnie++测试结果

图 5 为实验结果,由图可见,相对普通网络磁盘,异步镜像逻辑卷的性能下降分别为 61.3%(按字符顺序写)、21.8%(按块顺序写)、17.3%(重写)、10.1%(按字符顺序读)、4.6%(按块顺序读).其中,按字符顺序写的性能下降最大,这是因为写测试会频繁更改数据,而在异步镜像逻辑卷中,数据更新会被追踪并记录到增量卷,进而传输到远端卷中,这些操作会带来一定的计算资源和带宽资源的消耗.而按块顺序写的性能下降稍小,这是因为在 Bonnie++中,一个块大小为 8KB,比一个字节要大 3 个数量级还多,其写操作的聚合度更好,连续性更高,触发的更新追踪次数相对较少,故带来的额外开销较按字节写要小.重写操作过程中读操作不会触发数据更新,而且由于文件系统的缓存的缘故,其性能下降更小.两个读测试,没有数据更新,性能下降最少.

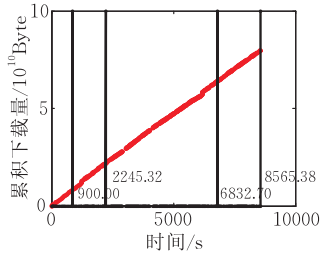
如果我们把上述各微测试的平均吞吐率再取平均,并以此作为异步镜像逻辑卷的一个性能度量,那么,我们能够得到如下结论:在异步镜像逻辑卷的实验中,各个微测试的混合 IO 吞吐率为 51946.8KB/s,而在普通网络磁盘的实验中,混合 IO 吞吐率达 63229.8KB/s;前者的 IO 性能较后者下降了 17.8%.

4.2　FTP Trace 测试

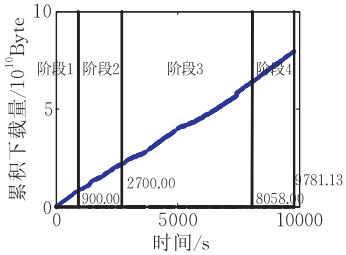
本测试中,FTP Trace 重放脚本运行在客户主机上,其运行流程如下:顺序读取 Trace 文件、依序建立会话、读取请求下载的文件、经由百兆以太网将文件传送至实验室的一台 PC、完成下载.由于重放时各个会话间并不等待,我们实际构造了一个对 IO 性能要求较高的压力测试环境.实验中,我们记录了每个 session 的起始时间、终止时间、下载数据量,通过观测累积数据下载量随时间的增长规律来衡量系统的整体性能.

首先,我们在场景 1 下做重放,即普通网络磁盘用以存放 FTP 资源,且全过程无故障.实验结果如图 6(a),可见整个 Trace 的重放需要 8565s.

然后,我们在场景 2 下做重放,将 FTP 资源置于异步镜像逻辑卷的保护之下,且人为加入节点失效.实验结果如图 6(b),整个过程可以分为 4 个阶段(为了便于比较,我们将图 6(a)也按照相应的会话编号做了 4 个阶段的划分):(1)本阶段 FTP 服务器工作正常.(2)在重放的第 900s,使"主/本地"节点失效,进入远端节点提供服务的阶段.从图中可见该阶段的总耗时,即 1800s,较场景 1 下的 1345.32s 更长,按平均下载速率,性能下降约为 25.3%.这是由于本阶段发生了节点失效,引发了节点切换,后续的 IO 请求不得不由远端节点来响应.(3)当第 2700s 时,触发恢复同步操作,以模拟本地节点的设备被修复并重新投入使用的事件,自此,进入第 3 阶段.由图可见,相对于场景 1 下耗时 4587.38s,本阶段在场景 2 下耗时增加到 5358s,性能下降约为 14.4%.(4)当到达第 8058s 的时候,恢复同步结束,系统进入第 4 阶段,本地节点又成为主节点,IO 请求的服务方又切换回本地节点.



(a)普通网络磁盘且无故障的场景



(b)异步镜像卷且发生故障的场景

图 6　FTP Trace 重放实验测试结果

由图 6,我们可以发现由于人为引入镜像机制和节点失效的故障,在阶段 2 和阶段 3,FTP 服务器依然能够提供服务,但性能也有所下降.整个重放时间由场景 1 下的 8565.38s 增加到场景 2 下的

9781.13s,也就是说,在我们的镜像系统的保护下,FTP 服务器并没有感觉到节点的故障,所花的代价仅仅是额外增加的约 14%的服务时间.

通过以上比较,我们可以得出结论:基于存储虚拟化的异步远程镜像系统成功地提供了对应用数据和服务的保护,在故障发生时以及灾后恢复过程中能够继续向客户主机提供服务,且服务质量的下降也被控制在合理的范围内.

5 结 论

本文论述了一种基于存储虚拟化的异步远程镜像系统的设计与实现.通过使用一种异步镜像逻辑卷作为被保护数据的容器,能够自动追踪、合并以及异步传输数据更新,透明地实现镜像功能且节省网络资源,同时不依赖于任何的底层设备和驱动.通过提出一种不间断服务的异步镜像协议,使服务能够在镜像主从节点间无缝迁移,支持灾难发生和灾后恢复时主从节点的自动切换和不间断服务,同时,该协议的执行不占用客户主机资源.设计了基于锁的镜像卷访问控制机制,既保证了数据的一致性,又能在此前提下尽可能提高读写并发性.实验结果证明,该系统具有高的可靠性,且由此带来的性能开销能够被控制在可接受的范围内.

参 考 文 献

[1] Veritas Volume Replicator Option by Symantec—A Guide to Understanding Volume Replicator, Engineering White Paper, Symantec, 2006

[2] Michail F, Angelos B. Clotho: Transparent data versioning at the block I/O level//Proceedings of the 21st IEEE Conference on Mass Storage Systems and Technologies (MSST 04). Greenbelt, Maryland, USA. IEEE Computer Society, 2004: 315-328

[3] Akshat V, Kaladhar V, Ramani R, Rohit J. SWEEPER: An efficient disaster recovery point identification mechanism//Proceedings of the USENIX File and Storage Technologies (FAST 08). San Jose, California, USA. USENIX Association, 2008: 297-312

[4] EMC SRDF: Zero data loss solutions for extended distance replication. EMC Corporation: Technical Report P/N 300-006-714, REV A03, 2009

[5] Hugo P, Stephen M, Mike F, Dave H, Steve K, Shane O. SnapMirror: File system based asynchronous mirroring for disaster recovery//Proceedings of the USENIX File and Storage Technologies (FAST 02). Monterey, California, USA. USENIX Association, 2002: 117-129

[6] Hitz D, Lau D, Malcolm M. File system design for an NFS file server appliance//Proceedings of the 1994 Winter USENIX Technical Conference. San Francisco, California, USA. USENIX Association, 1994: 235-245

[7] Minwen J, Alistair V, John W. Seneca: Remote mirroring done write//Proceedings of the USENIX Annual Technical Conference (USENIX 03). San Antonio, Texas, USA. USENIX Association, 2003: 253-268

XIANG Xiao-Jia, born in 1977, Ph. D. candidate. His main research interests include storage network, distributed file systems, disaster recovery, virtualization technology, and so on.

YU Hong-Liang, born in 1976, Ph. D., associate professor. His research interests include distributed system, disaster recovery, and so on.

Background

Nowadays, computerized data becomes more and more critical. Many kinds of data protection technologies are presented and have their own characteristics. Among them, mirror systems focus on seamless service switching and can survive even encountering site corruptions.

Unfortunately, existing mirror systems have some limitations. Firstly, dependency on the storage devices and low level utilities limits their flexibility. Secondly, existing mirror mechanisms always introduce negative overhead to the foreground hosts due to the resource contention.

In this paper, an asynchronous remote mirror system based on storage virtualization technology is presented. By utilizing logic volumes for mirroring, the system is independent of any storage devices and low level drivers. The logical volumes are also ease to extend and manage. An asynchronous mirror protocol which supports seamless service failover and a lock based concurrent access control mechanism are presented as well.

The authors have done researches on high reliability and disaster recovery for many years. They have taken part in many national research projects. By their studies, they have a deep understanding of backup, snapshot, synchronous/asynchronous mirror technologies and publish many papers in this area.

The authors also implement a prototype mirror system using aforementioned technologies. Experiment results show that it can tolerate site corruptions while keeping service online, costing only 14% more time.