高技术通讯 2010 年 第 20 卷 第 11 期:1101 ~ 1107

doi:10.3772/j.issn.1002-0470.2010.11.001

一种卷级连续数据保护一致点插入方法①

生拥宏② 汪东升 鞠大鹏 武 健

(清华大学计算机科学与技术系清华信息科学与技术国家实验室 北京 100084)

摘 要 形式化描述了应用层语义事件与卷级连续数据保护一致点插入的关系,提出了借助文件系统特定 IO 操作在正常 IO 流中插入一致点的算法。该方法提供的三种一致点插入技术分别能满足单个文件、目录和逻辑卷不同粒度的一致性恢复需要,同时也可以根据用户的定义弱化为传统的基于固定周期的快照备份方法。论文给出了系统的实现方法,评估了方法的有效性。实验表明,该方法提供了准确率较高的一致性恢复手段。

关键词 连续数据保护(CDP),一致性,备份

0 引 言

最近的研究调查发现,在很多行业中,数据丢失或者数据不可用造成的损失往往达到每小时上百万美元[1]。建立备份系统,提高计算机系统的可靠性与容灾能力,保证业务连续运行,是一个持续研究热点。备份技术也由复制、镜像、快照发展到连续数据保护(continuous data protection,CDP)。CDP 是一种数据的连续时间点的保护技术,可以在故障发生时实现到任意时间点的恢复,达到业务快速连续的作用,从根本上解决传统备份中低恢复能力和非精细时间策略的先天弱点。基于 CDP 的系统可划分为块级、文件级和应用程序级 3 类,其中块级 CDP 系统位于物理储存或逻辑卷管理层之上,独立于上层应用(文件系统、数据库系统及其他应用),具有很强的通用性,且适用于大数据量的作业系统。文献[2]提出了在存储控制器中实现持续数据保护的 4 种不同的架构,并从写性能和空间利用开销方面对其进行分析。文献[3]基于 TRAP 系统框架实现了一个块设备层次的持续数据保护驱动,并对空间占用开销和恢复时间进行了深入分析。文献[4]提出了基于 ATA 硬盘和 G 级以太网技术实现的综合的持续数据保护系统 Mariner。上述研究工作主要侧重于存储空间方面进行研究,在数据的一致性保证方面的研究不多。文献[5]基于系统事件,如软件更新、防火墙设置等自动插入有意义的时间点,在恢复时提供了快速查找方法。文献[6]通过分析文件系统元数据的写操作,插入一致的恢复点。文献[6]强依赖于文件系统的元数据结构,基于公开格式的文件系统 ext2 实现,而对像微软的 NTFS 则不适应。本文在分析传统快照保护不足的基础上提出了一种带一致点插入的卷级连续数据保护方法。该方法无需定义许多系统事件,仅通过捕获特定类型的 IO 就可插入一致的时间点。系统通过在正常的 IO 流中插入携带有意义的时间点特殊 IO,为用户数据的一致性恢复提供了辅助手段。通过用户自定义插入策略,该方法可以弱化为连续快照保护方法。与传统的单一快照和周期性快照相比,恢复精度更高、更准。

1 传统快照技术及卷级 CDP 一致点插入分析

快照技术能够实现数据的即时影像(point-in-time image),快照影像可以支持在线备份。为了降低快照所占用的存储空间,人们提出了写时拷贝(copy-on-write,COW)和写时重定向(redirect-on-write,ROW)快照技术。COW 快照只保存建立快照后被新的写操作覆盖的数据,而 ROW 快照只包括新的写操作数据。快照可以在磁盘阵列、文件系统、卷管理器、NAS 系统或者备份软件中实现。在逻辑卷实现数据保护具有很强的通用性,本文仅限于对卷级数据保护进行研究。在逻辑卷层次,数据由连续的数据块组成。每个数据块有固定的大小,并通过逻辑块地址(LBA)来标识。为研究方便,定义如表 1 所示的变量。

① 863 计划(2009AA01Z104)资助项目。
② 男,1977 年生,博士生,讲师;研究方向:体系结构,网络存储;联系人,E-mail: shengyonghong@gmail.com
(收稿日期:2009-09-11)

表 1 变量定义

A	逻辑卷 LBA 的地址集合
D	逻辑卷 LBA 对应的数据集合
R	A 和 D 的映射关系
F_t	t 时刻 A 至 D 的函数关系
$F_t(a)$	t 时刻 LBA 地址为 a 的数据
A'	A 的子集
F_t/A'	$F_t/A' = F_t \bigcap (A' \times D)$

为不失一般性,在时间轴上定义三个时间点:$i-1$,i 和 $i+1$。如图 1 所示。$i-1$,i 和 $i+1$ 分别表示数据起始时间、恢复时间点和当前时间。

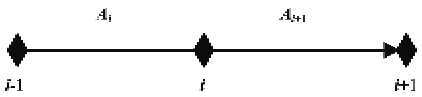

图 1 时间轴恢复点定义示意图

定义 1: A_i 为从时间点 $i-1$ 至 i 期间数据被覆盖的 LBA 的集合。

定义 2: A_i' 为从时间点 $i-1$ 至 i 期间数据发生变化的 LBA 的集合。

定义 3: F_i/A_{i+1} 为 i 至 $i+1$ 时刻的 COW 快照增量数据。

定义 4: F_i/A_i 为 $i-1$ 至 i 时刻的 ROW 快照增量数据。

假设 1: $F_i(\alpha)$ 和 $F_{i+1}(\alpha)$ 不相关。因此,对所有的 $b \in A$ 并且 $b \neq a$,$F_i(\alpha)$ 和 $F_i(b)$ 不相关。

由 COW 的定义可知,如果当前时刻为 $i+1$,COW 快照的增量数据从 i 时刻开始,则 i 时刻的存储数据 F_i 可表示为

$$F_i = (F_{i+1} - F_{i+1}/A_{i-1}) \bigcup F_i/A_{i+1} \quad (1)$$

式(1)描述了 COW 快照方式下可以通过 Undo 进行数据恢复。

由 ROW 的定义可知,如果当前时刻为 i,ROW 快照的增量数据从 $i-1$ 时刻开始,则 i 时刻的存储数据 F_i 表示为

$$F_i = (F_{i-1} - F_{i-1}/A_i) \bigcup F_i/A_i \quad (2)$$

式(2)描述了 ROW 快照方式下可以通过 Redo 进行数据恢复。

COW 快照和 ROW 快照可以恢复到历史时刻的指定时间点。但如果当前数据发生错误,则 COW 快照不能正确恢复;如果历史备份数据发生错误,则 ROW 快照不能正确恢复。详细形式化证明可参考文献[7]。

CDP 通过记录所有数据的变化,提供了任意时间点恢复功能,从根本上解决传统备份中低恢复能力和非精细时间策略的先天弱点。尽管 CDP 提供了任意点的恢复,但不是所有时间点恢复出的文件都是一致的。CDP 对每一个写的数据写入 CDP 后台存储并记录写入时间。

定义 5: CDP 日志中数据记录单元可用四元组 $B(a,i,l,c)$ 表示。其中,a 表示写操作的逻辑卷地址,i 表示时间戳,l 表示记录单元在 CDP 的存储位置,c 表示写入的数据内容。

定义 6: $B(a,i,*,*) \preccurlyeq B(b,k,*,*)$ 表示 $B(a,i,*,*)$ 是 $B(a,k,*,*)$ 的先导。即 i 时刻向地址 a 写入的数据块 $B(a,i,*,*)$ 和 k 时刻向地址 b 写入的数据块 $B(b,k,*,*)$ 之间没有其他数据块写入地址 a。形式化描述见公式

$$B(a,i,*,*) \preccurlyeq B(b,k,*,*) \rightarrow \nexists B(c,j,*,*)[(c=a) \wedge (i<j<k)] \quad (3)$$

其中 * 表示任意内容。

不同的 LBA 之间的数据逻辑关系由上层文件系统语义定义。文件系统之上可表示不同 LBA 之间数据逻辑关系的有个单文件、目录和逻辑卷。

定义 7: i 时刻单个文件 S_i 可用二元组 (M_i,D_i) 表示。其中 M_i 表示 S_i 的元数据,即 S_i 包含的 LBA 的集合。D_i 表示 i 时刻单个文件 S_i 包含的数据集合。

图 2 表示某单文件 S,i 和 $i+1$ 时刻 M 和 D 的关系示意图。

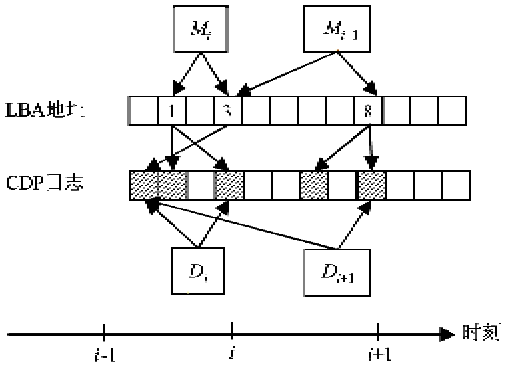

图 2 时刻 i 和 $i+1$ 单文件 S 的 M 和 D 关系图

图 2 表示,$i-1$ 至 i 期间逻辑地址为 1 的数据块变化 2 次,逻辑地址为 3 的数据块未发生变化。i 时刻文件 S 由逻辑地址为 1 和 3 的数据块组成。i 至 $i+1$ 期间逻辑地址为 8 的数据块变化 2 次。$i+1$ 时刻文件 S 由逻辑地址为 3 和 8 的数据块组成。

对于给定的时刻 t,文件 S 的恢复过程由以下步骤组成:

(1)找离时刻 t 最近的文件 S 的元数据 M_t;

(2)确定元数据 M_t 的最后一个更新数据块,设为 $M_t(b,k,*,*)$;

(3)根据元数据 M_t,在 CDP 日志中找出满足条件的块数据集合 D_t。$B(a,i,*,*)$是否是 D_t 中的元素由下列判断条件公式

$$B(a,i,*,*)\in D_t \rightarrow (a\in M_t)\wedge B(a,i,*,*)\preccurlyeq M_t(b,k,*,*) \tag{4}$$

确定。

卷级 CDP 的先天缺陷是不能保证任意时刻恢复的文件是一致和有效的。文件是否一致和有效与上层的应用紧密相关。恢复的文件内容由离时刻最近的文件 S 的元数据 M_t 确定。

定义 8: $\tau(M_t)=0$ 表示由 M_t 恢复出的文件是一致的。

如果能在时间轴上人为插入一致的检查点(Checkpoint),则有助于用户提高文件恢复的一致性。一般而言,一致检查点的插入与特定的文件操作相关联,如文件的关闭、删除和重命名等。

定义 9: E 为插入一致点的触发事件集合,CP 为与相关联触发的一致检查点。$CP(E_i)$为事件 E_i 触发的 CP。

t 时刻一致点触发事件 E_i 触发的 CP 是否能保证文件恢复的一致性,与 CP 的实际写入时间 t_1 和元数据 M_t 的实际更新时间 t_2 有关。如果能保证一致点在元数据更新后插入,而且在一致点插入前没有数据写入,则可保证恢复的文件是一致的,约束条件可由公式

$$\tau(M_t \mid CP(E_i))=0 \rightarrow M_t(b,t_2,*,*)\preccurlyeq CP(*,t_1,*,*)\wedge \nexists B(c,j,*,*)[(c\in M_t)\wedge(t_2<j<t_1)] \tag{5}$$

表示。

由于 CP 的插入独立于文件 S 的写操作,在实际中很难保证 $\tau(M_t \mid CP(E_i))=0$ 有时会出现 CP 会在 $M_t(b,t_2,*,*)$前写入 CDP 日志。在事件 E_i 发生后适当延时 Δt 后写入 CP 会提高文件恢复一致的成功率,因此 CP 提供的恢复点能保证文件的一致性的概率可由公式

$$P(\tau(M_t \mid CP(E_i))=0)=P(M_t(b,t_2,*,*)\preccurlyeq CP(*,t_1,*,*)\wedge \nexists B(c,j,*,*)[(c\in M_t)\wedge(t_2<j<t_1)] \mid \Delta t \tag{6}$$

表示。

上面形式化描述了一致点插入与单个文件恢复一致性的关系。对目录而言,同样可以在目录语义事件上关联插入一致点动作。

2 卷级 CDP 一致点插入实现

2.1 系统结构描述

如图 3 所示,系统由客户端和 CDP 服务端组成。客户端通过在文件系统的上层和下层分别插入一致点捕获驱动(CCD)和镜像驱动(MD)分别实现一致点的捕获和块级数据的远程镜像。被保护对象所有变化的数据通过 iSCSI 协议发送至 CDP 服务器端。在传送变化的块数据前,被保护的本地逻辑卷数据被完全镜像至 CDP 服务器端。本方案设计中,被保护卷的完整镜像数据和卷变化数据独立于生产数据存放,因此可以克服上述 COW 和 ROW 由于历史数据或当前数据错误而不能正确的不足。

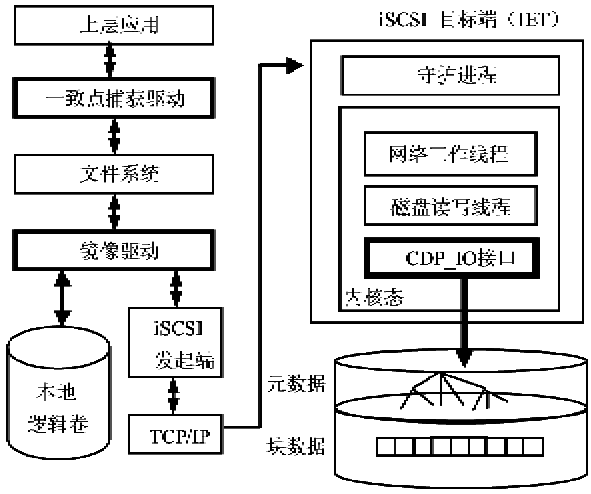

图 3 CDP 组成结构图

CDP 服务器基于开源软件 IET(iSCSI Enterprise Target)实现,通过引入 CDP _ IO 接口实现了变化数据在裸盘上的存储。元数据记录了原始镜像文件、索引文件和一致点等信息。变化的块数据组织成段结构后存储至裸盘。CDP 服务器提供的历史镜像查看视图可以在当前保护不中断的前提下满足用户历史任意点视图的查看和历史文件拷贝功能,具体实现细节参见参考文献[8]。

2.2 一致点 IRP 自动插入机制

IRP(I/O Request Package)是 Windows 内核中的一种非常重要的数据结构。上层应用程序与底层驱动程序通信时,应用程序会发出 I/O 请求,操作系统将相应的 I/O 请求转换成相应的 IRP,不同的 IRP 会根据类型被分派到不同的派遣例程中进行处理。文件 I/O 的相关函数,例如 CreateFile、ReadFile、WriteFile、CloseHandle 等分别会引发操作系统产生 IRP _ MJ _ CREATE、IRP _ MJ _ READ、IRP _ MJ _ WRITE、IRP _ MJ _ CLOSE 等不同类型的 IRP,这些 IRP 会被传送到驱动程序的相应派遣例程中进行处理。

本系统镜像驱动截获了每一个写操作 IRP,并把写内容通过 iSCSI 协议发送到 CDP 服务器存储。系统支持任意时间点的恢复,但如果把截获的每一个 IRP 产生的块数据都作为恢复点,恢复时可能出现文件无效的现象。

文件系统实现了文件级逻辑地址到卷级逻辑地址的转换。如文件系统中 1 个 IRP 需要将 8k 字节内容从文件偏移为 20480 字节处写如入文件 A。如果文件系统块大小为 4k 字节,则经过文件系统处理后,会产生 2 个写 IRP 发送给镜像驱动,每个 IRP 中指定了 4k 字节写入逻辑卷的偏移地址。由本文第 1 节的假设 1 可知,块数据之间是不相关的,而且在镜像驱动层捕获的块数据没有上层语义信息,即在镜像驱动捕获的 IRP 中没有描述该块数据对应是上层哪个文件的信息。语义信息只能在文件系统之上捕获。本系统中通过在文件系统之上插入一致点捕获驱动来捕获语义信息,然后将一致点信息封装成特殊的 IRP 发送给镜像驱动,亦即在普通的 IO 流中插入了一致点信息的 IO 块数据。系统结构如图 4 所示。

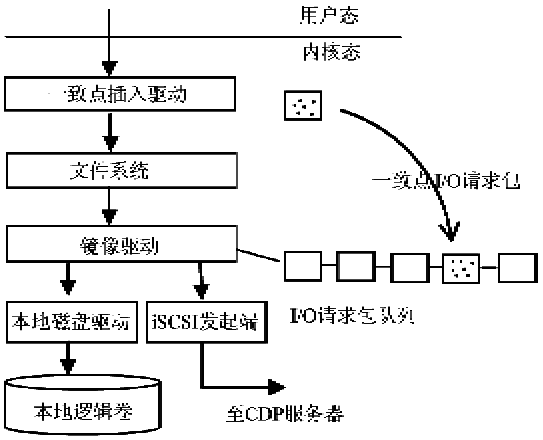

图 4 一致点 IO 插入示意图

在插入一致点 IRP 时,有几点设计原则:

(1)能区分正常写操作 IRP 和一致点 IRP;

(2)选择正确的插入时机;

(3)一致点 IRP 只能发送到远程 CDP 服务器,不能写入本地逻辑卷。

本系统的一致点 IRP 设计为写类型的 IRP,写地址为逻辑卷的 0x1CE 处,即写到主引导扇区(MBR)的第 2 个磁盘分区表处。一个 MBR 记录包含 4 个磁盘分区信息。由于 CDP 服务器通过一个逻辑单元号(logical unit number,LUN)和本地被保护的逻辑卷进行镜像。在实际使用中,CDP 的一个 LUN 只用了第一个分区,因此 CDP 服务器镜像 LUN 的 MBR 的第 2-4 个分区的信息没有实际意义。1 个分区记录包含 16 个字节,因此本系统中可用 MBR 的 3 个空闲分区信息,共计 48 个字节来传送和表达一致点的信息。

镜像驱动维持一个写 IRP 的队列,通过异步方式将写 IRP 发送至 CDP 服务器。一致点 IRP 只插入写往 CDP 服务器的 IRP 队列,对写入本地逻辑卷的 IRP 没有任何影响。

一致点 IRP 的插入时机根据用户的策略定义。本系统设计中,一致点 IRP 分为 3 类:

(1)弱一致 Checkpoint IRP:在单个文件关闭、删除和重命名等操作时写入,一致性的支持粒度为文件;

(2)强一致 Checkpoint IRP:在被保护的目录所有打开的文件都关闭时写入,一致性支持粒度为目录;

(3)强制一致 Checkpoint IRP:按用户定义的策略强制性写入,一致性支持粒度为逻辑卷。

弱一致和强一致 Checkpoint IRP 插入时间如图 5所示。

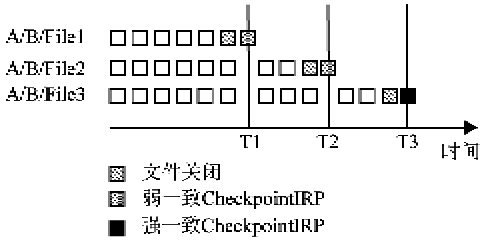

图 5 弱一致和强一致插入时间示意图

一致点插入驱动可以根据用户的策略定义,可以对指定的目录夹和指定的文件类型进行一致性辅助决策支持。对文件类型进行过滤设置能避免一致

点插入驱动对临时文件的额外处理开销。例如,微软 Office Word 文档编辑软件在编辑文档时会产生临时文件,Window 操作系统也会在系统中产生虚拟内存交换文件。捕获这些临时文件的关闭和删除操作意义不大。

弱一致 Checkpoint IRP 通过分析 IRP _ MJ _ CLOSE 和 IRP _ SET _ INFORMATION 两种类型的 IRP 产生。上层应用在关闭文件句柄时调用系统调用 CloseHandle 方法,CloseHandle 向文件系统发送 IRP _ MJ _ CLOSE 类型的 IRP。一致点插入驱动通过分析 IRP _ MJ _ CLOSE 类型 IRP 中关联的文件信息判断是用户感兴趣的文件类型或文件保护目录,如果是则插入一个封装弱一致 IRP 后随同上层 IRP _ MJ _ CLOSE 类型的 IRP 发往文件系统。对 IRP _ SET _ INFORMATION 类的 IRP,一致点捕获驱动只对次功能类型为删除、重命名和修改属性的 IRP 产生弱一致 Checkpoint IRP,而对次功能类型为剪切和修改文件指针的操作不产生弱一致 Checkpoint IRP。系统中主要的 IRP 类型和一致点捕获驱动关心和处理的 IRP 说明见表 2。

表 2　IRP 类型和一致点捕获关联表

操作	来源	上层调用	捕获
IRP _ MJ _ CREATE	创建、打开文件	CreateFile	一致点捕获驱动处理
IRP _ MJ _ CLEANUP	在关闭句柄时取消悬挂的 IRP	CloseHandle	不处理
IRP _ MJ _ CLOSE	关闭文件句柄	CloseHandle	一致点捕获驱动处理
IRP _ SET _ INFORMATION	删除、重命名、修改属性、剪切、修改文件指针。具体操作在 IRP 的次功能代码中定义	SetFileLength	一致点捕获驱动部分处理
IRP _ MJ _ WRITE	写入数据	WriteFile	镜像驱动处理
IRP _ MJ _ FLUSH _ BUFFERS	写输出缓冲区或丢弃输入缓冲区	FlushFileBuffers	不处理

弱一致 Checkpoint IRP 指示了文件的一致点,而强一致 Checkpoint IRP 则指示了目录的一致点。强一致 Checkpoint IRP 通过计数器实现。对用户指定的被保护目录,上层应用打开目录下的文件时会往文件系统发送 IRP _ MJ _ CREATE 类型的 IRP。与该目录关联的计数器加一,在关闭某个文件时计数器减一。如果计数器为零,则表明该目录下所有打开的文件全部关闭。此时,一致点捕获驱动插入一个封装强一致信息的 Checkpoint IRP。

强制一致 Checkpoint IRP 则根据用户预先定义的策略生成。例如,用户要求 1 小时生成一个强制一致视图,则系统每个 1 小时插入一个强制一致 Checkpoint IRP。系统插入强制一致 Checkpoint IRP 时调用操作系统的 Flush 命令强制将文件系统中的缓存数据刷新到磁盘上。通过插入强制一致 Checkpoint IRP,系统只能提供 Crash-consistency 级别的数据一致性保证,而不能提供类似于 Windows 卷影复制服务提供的应用级一致(Application-level consistency)的快照视图。

2.3　CDP 服务器对 Checkpoint IRP 的利用和处理

客户端镜像驱动保证了 IRP 严格按时间顺序将块数据发送到 CDP 服务器端。CDP 服务器端在接受到 IRP 时,按以下流程进行处理:

(1)分析是普通写 IRP,还是一致点 IRP。

(2)如果是普通写操作 IRP,提取写地址偏移和写内容,结合服务器当前时间封装成最小日志单元格式,转移至缓存等候处理。

(3)如果是一致点 IRP,首先对缓存中等候写入的最小日志单元进行去除重复写处理,即在两个一致点时间间隔内重复写往同一地址的数据只保留最后一份,然后启动将缓存数据写入日志,写入成功后将一致点信息写入特定的一致点索引文件。

一致点 IRP 到达是缓存数据写入日志的触发条件,但不是充分条件。本系统设计中引入了 3 种写触发条件:基于时间的触发,基于缓存空间的触发和一致点 IRP 触发。为安全起见,当缓存数据在内存中滞留时间,或缓存使用率到达预设的阈值后也触发写日志事件。

3　测试和评价

系统基于自主研发的连续保护系统 TH-CDP 进行实验,TH-CDP 总体实现见参考文献[8]。系统实验环境配置如表 3 和表 4 所示。

表 3 CDP 服务器配置

Ubuntu kernel	2.6.18
iSCSI Enterprise Target	0.4.16
硬盘	Single SATA 250GB

表 4 CDP 客户端配置

Windows iSCSI initiator	iSCSI initiator 2.04
IOzone	IOzone 3.321
硬盘	Single SATA 250GB

3.1 双驱动对被保护系统的写性能影响测试

一致点捕获驱动和镜像驱动分别安装在文件系统的上层和下层，所有写入被保护逻辑卷的 IRP 需经过这两层驱动(简称“双驱动”)。由于引入了额外的处理，被保护逻辑卷的写入速度会受到一定的影响。

IOzone 是一个文件系统的 benchmark 工具，可以测试不同的操作系统中文件系统的读写性能。因为双驱动仅对写操作有影响，因此在 IOzone 中仅测试了双驱动对文件系统写性能的影响。测试块大小设置为 64k 至 8192k。测试分三种情况：(1)不安装双驱动；(2)安装双驱动，但驱动不插入 Checkpoint IRP，且不发送镜像 IRP 至 CDP 服务器，此时驱动相当于空负载；(3)双驱动完全工作。测试结果如图 6 所示。测试结果表明，驱动空负载对系统的写性能有 2% ~3% 的性能影响，驱动完全工作，对系统的写性能有 5% ~6% 的影响。

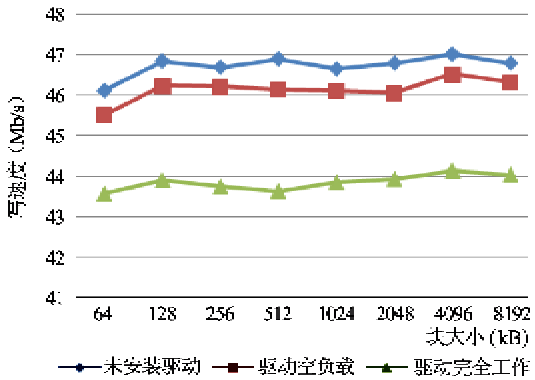

图 6 驱动对系统写 IO 性能影响测试结果

3.2 一致点恢复正确性测试

通过 Word 字处理软件对文档的操作来测试一致点恢复的正确性。本实验编写了一个 VBA 测试程序，该程序自动打开特定目录的单一或同时打开多个 Word 文档，写入特定的内容后关闭。打开文件、写入内容、删除内容和关闭操作循环执行 5000 次后，测试程序按 TH-CDP 提供的一致点打开文件验证内容是否原先写入的特定内容。如果打开文件失败或内容不一致，则提供的一致点不准确。实验表明，一致点的准确性与插入 Checkpoint IRP 的时延 D 有关系。如果在一致点插入驱动捕获到一致点 IRP 后立即插入 Checkpoint IRP，亦即 D = 0，此时恢复正确率最低。

如果适当延时一段时间后插入 Checkpoint IRP，则恢复的正确性得到提高。这主要是由于文件系统对普通写的 IRP 有缓存作用，可能出现后写入的 Checkpoint IRP 比普通写 IRP 先到达底层镜像驱动。通过适当加入 Checkpoint IRP 写延时可以提供一致点的准确性。系统实验数据如表 5 所示。

表 5 恢复正确率测试结果

类别	测试点	恢复正确率		
		D = 0ms	D = 0.5ms	D = 1ms
弱一致	5000	82.4%	90.2%	94.3%
强一致	1000	78.2%	84.3%	90.5%
强制一致	1000	85.2%	90.3%	95.2%

实验数据表明，适当延时后插入 Checkpoint IRP 会提高恢复点的准确率。但也不是越长越好，这是因为很可能在延时的过程中用户又打开了文件。实验还表明，强制一致性类型的插入点的恢复准确率高于其余两种一致点插入方式，这是因为强制一致将缓存数据刷新到物理存储上后再插入一致点。

3.3 存储空间节省测试

在两个一致点之间如果上层应用对同一数据块多次写入数据，TH-CDP 只将最后一次块数据写入 CDP 存储池。存储空间的减少与上层具体应用相关。例如，对一个文件下载的应用则不会减少存储空间，因为从下载文件的创建到关闭的操作过程中没有文件改写。本系统对一个 Word 文件打开后，执行了多次写入和删除操作后关闭文件。通过查看最终实际写入的数据块数量验证了系统对同一地址的重复写仅将最后一次数据块写入了存储池。

4 结 论

卷级连续数据保护在提供足够精细恢复粒度的同时带来了恢复一致性的难题。本文通过在文件系统之上捕获关键数据操作信息，然后在正常的数据

流中插入一致点信息。该方法提供的弱一致、强一致和强制一致等3种手段分别能分别满足单文件、目录和逻辑卷不同粒度一致性恢复的需要。实验结果表明,80%以上的数据通过一致点辅助决策点恢复后是一致和正确的。下一步的研究重点是进一步提高一致点恢复的准确性和解决数据库恢复的一致性难题。数据库操作与普通的文件系统操作存在很大的不同,一个典型的数据库应用在打开数据库后直至应用停止才关闭数据库。在卷级保证数据库的一致性恢复不能采用文中所述的捕获IO的语义信息,而应在应用层捕获数据库交易的语义信息,这是我们下一步研究的重点。

参考文献

[1] Keeton K, Santos C, Beyer D, et al. Designing for disasters. In: Proceedings of the 3rd USENIX Conference on File and Storage Technologies, San Francisco, USA, 2004.59-72

[2] Laden G, Ta-shma P, Yaffe E, et al. Architectures for controller based CDP. In: Proceedings of the 5th USENIX Conference on File and Storage Technologies, San Jose, USA, 2007. 107-121

[3] Yang Q, Xiao W, Ren J. TRAP-Array: a disk array architecture providing timely recovery to any point-in-time. In: Proceedings of the International Symposium on Computer Architecture, Boston, USA, 2006. 289-301

[4] Lu M, Lin S, Chiueh T. Efficient logging and replication techniques for comprehensive data protection. In: Proceedings of the 24th IEEE Conference on Mass Storage Systems and Technologies, San Diego, USA, 2007.171-184

[5] Verma A, Voruganti K, Routray R, et al. SWEEPER: an efficient disaster recovery point identification mechanism. In: Proceedings of the 6th USENIX Conference on File and Storage Technologies, San Jose, USA, 2008.283-296

[6] Lu M, Chiueh T, Lin S. An incremental file system consistency checker for block-level CDP systems. In: Proceedings of the 2008 Symposium on Reliable Distributed Systems, Napoli, Italy, 2008.132-140

[7] Xiao W, Ren J, Yang Q. A case for continuous data protection at block level in disk array storages. *IEEE Transactions on Parallel and Distributed Systems*, 2009, 20(6): 898-911

[8] Sheng Y, Wang D, He J, et al. TH-CDP: an efficient block level continuous data protection system. In: Proceedings of the 2009 IEEE International Conference on Networking, Architecture, and Storage, Zhangjiajie, China, 2009. 395-404

An efficient consistent point inserting method for block level continuous data protection

Sheng Yonghong, Wang Dongsheng, Ju Dapeng, Wu Jian

(Department of Computer Science and Technology, Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084)

Abstract

This paper presents a formal description on the relationship between semantic of application and consistent checkpoints of continuous data protection and proposes an algorithm of inserting consistent checkpoints into normal IO by analyzing specific IO operations. The proposed three types of consistent checkpoint inserting methods can satisfy the consistence of single file, directory and logical volume respectively. The method can not only provide any point in time data recovery but also provide periodical snapshot backup under user definition. The implementation and experiment are described in the paper. The extensive experiment results demonstrate that the proposed methods can achieve a high accuracy in consistent restoration of block level continous data protection.

Key words: continuous data protection (CDP), consistency, backup