第 34 卷 第 8 期 计 算 机 工 程 2008 年 4 月
Vol.34 No.8 Computer Engineering April 2008
·博士论文· 文章编号：1000—3428(2008)08—0041—03 文献标识码：A 中图分类号：TP333

基于流命令的 SCSI 目标端设计

康剑斌，汪海山，屠升平，贾惠波
(清华大学精密测试技术及仪器国家重点实验室，北京 100084)

摘 要：介绍一种基于 SCSI 流命令的 SCSI 的目标端，可以将接收到的流命令转换成针对 SCSI 磁盘的块传输命令。为了得到更好的传输性能，在该目标端中实现了缓存机制。分析了缓存对传输速度的影响，建立了 SCSI 目标端的传输模型，给出了仿真结果和实验结果。
关键词：SCSI 目标端；缓存；存储

Design of SCSI Target Based on Stream Command Set

KANG Jian-bin, WANG Hai-shan, TU Sheng-ping, JIA Hui-bo
(State Key Lab for Precision Test Technology and Instrument, Tsinghua Unversity, Beijing 100084)

【Abstract】SCSI is a mature protocol and used widely in storage system. This paper presents an implementation of SCSI target based on SCSI stream command set. This target receives stream command and converts it to block command. A cache mechanism is employed for better transfer performance. Some experiments on the influence of cache are carried out and the results are analyzed.
【Key words】SCSI target; cache; storage

1 概述

由于信息的爆炸性增长，存储系统的性能和可靠性也越来越受到重视。SCSI 协议广泛应用于高端的存储系统，如 iSCSI 存储系统。光纤协议也支持通过 SCSI 协议进行数据传输。SCSI 传输需要一个发起端(initiator)和一个目标端(target)，目标端接收发起端的命令并执行相应的操作。SCSI 磁盘就是一种常见的 SCSI 目标端。由于 SCSI 协议高性能、高可靠性的特性，因此许多存储系统的传输协议基于 SCSI 协议实现。

文献[1]介绍了一种用于存储局域网络(Storage Area Network, SAN)的 SCSI 目标端。该系统基于 Linux 内核实现，可以通过 SEP(SCSI Encapsulation Protocol, SCSI 封装协议)和 iSCSI 协议从 SAN 网络中的发起端获取 SCSI 命令和数据，并传递给 SCSI 子系统。该系统的缺点是，它只能用 SCSI 设备作为存储媒介。

文献[2]介绍了一种基于以太网的 SAN 协议，通过 Ethernet 协议而不是 TCP/IP 传输 SCSI 命令和数据包。这种系统避免了 iSCSI 中 TCP/IP 协议处理的开销，又具备了 SCSI 的高可靠性和通用性。这种协议的缺点是缺乏 TCP 传输的可靠性，必须自己处理丢包或错误的情况。

文献[3]介绍了一种多协议转换器的实现，可以处理多种 SCSI 协议，如光纤、iSCSI 和并行 SCSI，内部支持将面向流的 SCSI 命令转化为块设备的命令。该系统面向复杂的存储系统，可以方便地对存储系统进行管理，易于实现设备的虚拟化，但在一些对体积要求比较严格的领域，如野外勘探等，则显得过于庞大。

本文介绍一种基于并行 SCSI 接口实现的目标端。该目标端接收 SCSI 流命令，转换成块命令，并实现后端块设备(如 SCSI 磁盘或 RAID 阵列)的读写。为了提高系统的传输性能，该目标端提供了缓存机制，可将来自启动端的多个小数据块组织成较大的数据块，再集中传输到后端设备。本文建立了 SCSI 目标端的模型，分析了影响 SCSI 目标端读写速度的因素，并给出了实验结果。

2 SCSI 目标端设计和实现

2.1 SCSI 目标端传输模式

如图 1 所示，SCSI 目标端由以下 3 个部分构成：

(1)SCSI 接口。从发起端接收 SCSI 数据和命令。

(2)目标端中间层。SCSI 目标端核心模块，进行 SCSI 命令解析和协议转换。

(3)SCSI 模块和磁盘。目标端的存储介质，用于存储数据。

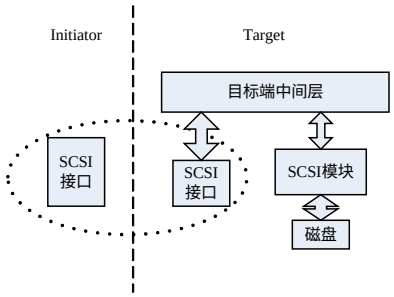

图 1 SCSI 目标端结构

从图 1 中可看出，SCSI 目标端有两个数据总线，一个是与发起端连接的 SCSI 总线，另一个是与存储设备连接的数据总线。系统的传输性能由这两个总线的吞吐量决定。SCSI 目标端的工作模式如下：响应发起端的命令、执行命令、向发起端汇报状态。以‘写’命令为例，如图 2 所示，传输过程如下：

(1)目标端接收发起端发出的写数据命令。

(2)目标端中间层响应这个命令并切换到数据传输状态。

作者简介：康剑斌(1982－)，男，博士研究生，主研方向：虚拟存储，网络化光盘库；汪海山，博士研究生；屠升平，硕士研究生；贾惠波，教授、博士生导师
收稿日期：2007-04-26 **E-mail：**kjb00@mails.tsinghua.edu.cn

(3)目标端开始接收来自发起端的数据。

(4)目标端将接收到的数据写入磁盘。

(5)目标端接收完数据，并向发起端返回完成状态。

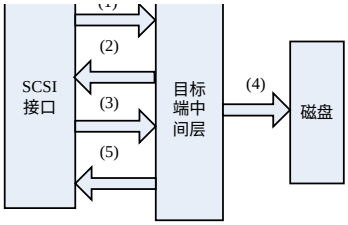

图 2　SCSI 目标端传输模式

2.2　传输块大小对系统性能的影响

在文献[2,4]中，用下面两个指标来衡量 SCSI 目标端的性能：

(1)吞吐量，即单位时间能传输的数据量。

(2)平均响应时间，即从发起端发出请求到目标端完成该请求所需要的时间。

由于本文实现的是流命令的 SCSI 目标端，主要操作集中在连续的数据读取和写入，很少发生寻址和查询，因此衡量该目标端性能的主要指标是系统的吞吐量。下面主要分析影响系统吞吐量的因素。

SCSI 传输以数据块为单位。块的大小由启动端在命令中指定。仍以图 2 为例，假设每次传输固定块大小为 S，目标端采用的存储设备也是 SCSI 设备。从发起端发出 SCSI 命令到目标端响应这个命令并进入数据传输时间为 t_1+t_2 (即图 2 中的阶段(1)、阶段(2))；发起端向目标端传输数据时间为 t_3 (即图 2 中的阶段(3))，则有

$$t_3 = kB$$

其中，k 为传输单位数据所需的时间，在接口速度固定的条件下，假定为常数；B 为单次传输数据块大小。

目标端完成数据传输，向发起端返回完成状态所需时间 t_5 (即图 2 中的阶段(5))。

对磁盘来说，目标端相当于发起端，仍需经过阶段(1)~阶段(5)。因此，目标端向磁盘写入数据所需时间为

$$t_4 = t_1 + t_2 + t_3 + t_5 + t_w$$

其中，t_w 为磁盘写入大小为 B 的数据所需的时间，可令

$$t_w = k_2 B$$

其中，k_2 假定为常数。

根据上面的分析，可得发起端传输大小为 B 的数据块需要时间为

$$t_b = t_1 + t_2 + t_3 + t_4 + t_5 = 2t_1 + 2t_2 + 2t_5 + 2kB + k_2 B$$

其中，$t_1+t_3+t_5$ 为协议传输的开销，假定为恒定值，记为 t_p。则传输大小为 B 的块所需时间为

$$t_b = 2t_p + 2kB + k_2 B$$

传输速度 S 为

$$S = \frac{B}{t_b} = \frac{B}{(2k+k_2)B + 2t_p} \tag{1}$$

从式(1)中可看出，块越大，协议开销在传输中所占的比例越小，传输速度越块。当块大小趋于无穷大时，极限传输速度为

$$S = \frac{1}{2k+k_2}$$

2.3　缓存的实现

从上面的分析可以看出，提高数据块的大小可以极大地提高系统传输性能。但由于块大小是发起端在命令中指定的，因此 SCSI 目标端无法随意改变传输的块大小。为了尽量减小块大小对传输性能的影响并提高 SCSI 带宽利用率，一般需要在目标端采用缓存机制。文献[5]介绍了一种用于 iSCSI 的缓存机制，将 iSCSI 的性能提高了 53%~78%。文献[4]也在实现的目标端采用了缓存机制，对数据进行索引并将 SCSI 请求进行排队和合并以提高目标端性能。

考虑到流传输的特性，笔者构造了一个缓存管理器 RAM Cache。该缓存管理器将接收到的小块数据放到一个 RAM Cache 的缓存池，然后将多个小块的数据合并成大块的数据，一次传输到磁盘上。由于流传输很少对数据进行查询，因此该缓存管理器不需要对数据进行索引。

加入缓存机制后，传输过程如图 3 所示。其中：

(1)目标端接收发起端发出的写数据命令。

(2)目标端中间层响应这个命令并切换到数据传输状态。

(3)目标端开始接收来自发起端的数据。

(4)目标端将接收到的数据写入 RAM Cache。

(5)目标端接收完数据，并向发起端返回完成状态。

(6)当缓存占用率超过一定值时，RAM Cache 将缓存中的数据转存到磁盘上。

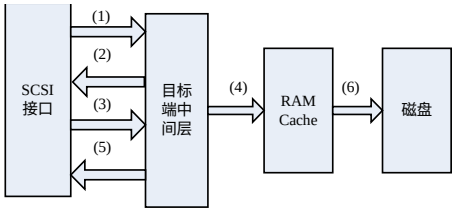

图 3　带缓存的 SCSI 目标端传输模式

加入缓存管理器 RAM Cache 后，由于目标端接收到数据后并未立即传输到磁盘上，在缓存未写满的情况下，一次传输所需时间为

$$t_{\text{empty}} = t_1 + t_2 + t_3 + t_5 + t_4 = t_p + kB$$

其中，t_4 为目标端将数据放入缓存的时间，可忽略，其余变量含义同上。

在缓存已写满的情况下，目标端需先用块大小为 B_2 ($B_2 > B$)的方式将 n 个数据块传输到磁盘上(即图中的(6))，此时，传输时间为

$$t_{\text{full}} = t_1 + t_2 + t_3 + t_4 + t_5 + t_6 = 2t_p + kB + nkB_2 + nk_2 B_2$$

其中，$t_6 = t_p + kB_2 + k_2 B_2$ 为一次传输大小 B_2 到磁盘所需的时间。则平均每次传输所需时间为

$$t_b = \frac{t_{\text{empty}}(nB_2/B) + t_{\text{full}}}{1 + nB_2/B} = \frac{nB_2(t_p + 2kB + k_2 B) + 2t_p B + kB^2}{nB_2 + B}$$

在大数据量的持续传输中，由于磁盘的读写速度远小于 SCSI 传输速度，缓存大部分时间处于写满状态，因此可认为 t_b 即为平均每次传输所需时间。可得传输速度为

$$S = \frac{(nB_2 + B)B}{nB_2(t_p + 2kB + k_2 B) + 2t_p B + kB^2} \tag{2}$$

3　实验结果

该 SCSI 目标端是作为虚拟磁带库(Virtual Tape Library, VTL)的一部分被实现的，用于户外高性能记录系统，对数据

的写入速度要求很高。因此，对该目标端的测试主要集中在写入速度的测试。表 1 是 SCSI 目标端和 SCSI 发起端所用的硬件配置。

表 1　发起端和目标端的硬件配置

	发起端	目标端
CPU	Intel Xeon	Intel IQ80331
内存	1 GB	512 MB
操作系统	Linux 2.6.14	Linux 2.6.15.4

在实验中，采用不同的块大小，对有缓存和无缓存两种情况分别进行了测试。实验结果如下所示：

(1) 当系统不采用缓存时，测得 $t_p = 0.153$ ms；$k = 7.287\times10^{-3}$ ms/KB；$k_2 = 1.000\times10^{-2}$ ms/KB。

根据式(1)可计算得到不同块大小下的理论传输速度。图 4 是理论计算的传输速度和实测传输速度的比较。

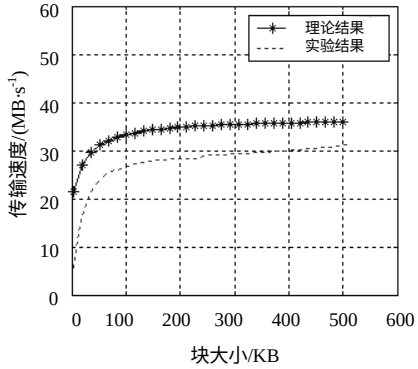

图 4　无缓存的理论和实测传输曲线

(2) 当系统采用缓存机制时，测得 $t_p = 0.153$ ms；$k = 3.21\times10^{-3}$ ms/KB；$k_2 = 1.000\times10^{-2}$ ms/KB。

取 n=1，根据式(2)可计算得到不同块大小下的理论传输速度。图 5 是理论计算的传输速度和实测传输速度的比较。

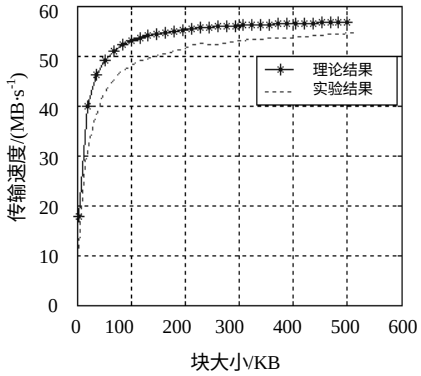

图 5　有缓存的理论和实测传输曲线

(3)图 6 是在不同块大小下，实测的有缓存与无缓存传输速度比较。

由图 4~图 6 可见，理论曲线与实验曲线的变化趋势是一致的。由于模型忽略了系统进程切换、内存管理的开销，因此理论传输速度大于实际的传输速度。对于无缓存的情况，由于进程切换、内存管理发生在每次数据块传输中，因此，它的开销大大影响了目标端对命令的响应速度，导致实际传输速度与理论值的差别增大；对于有缓存的情况，进程切换、内存管理可以和数据块传输并行(例如，在等待磁盘读写时可同时接收下一个 SCSI 命令)，对传输速度影响较小。进程切换、内存管理较为复杂，且与具体实现相关，这部分开销的计算需要进一步的工作。

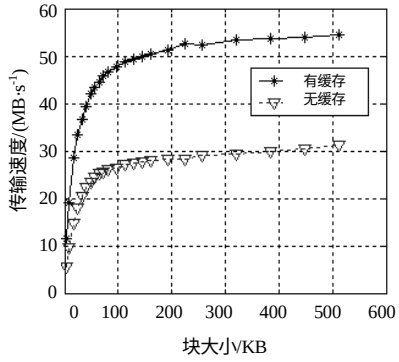

图 6　实测传输速度曲线

从实验结果可看出，采用缓存的传输速度远大于未采用缓存的传输速度，一方面由于将小块数据合并成大块减小了协议的开销；另一方面由于目标端的外部 SCSI 总线(接发起端)和内部 SCSI 总线(接磁盘)可以并行传输，提高了总线的利用率。

由图 6 可见，当块大小大于 128 KB 时，块大小对传输速度的影响大大减小。在内存较为紧张的场合，采用 64 KB 或 128 KB 的传输块大小是较好的选择。

4　结束语

本文介绍了一种基于流传输协议的 SCSI 目标端设计与实现，并分析了目标端的传输模式。该目标端通过 SCSI 目标端中间层，将接收到的流传输命令转换成针对磁盘的块传输命令。

针对流传输的特性，本文设计了一种缓存机制以提高传输性能，并建立了该缓存的模型。从理论计算和实验结果可看出，合适的缓存机制可以大大提高目标端的性能。

参考文献

[1] Palekars S. Design and Implementation of a Linux SCSI Target for Storage Area Networks[C]//Proceedings of the 5th Annual Linux Showcase & Conference. Oakland, CA, USA: [s. n.], 2001.

[2] Wilson Y H W. Design and Development of Ethernet-based Storage Area Network Protocol[C]//Proceedings of the 12th IEEE International Conference on Networks. [S. l.]: IEEE Press, 2004.

[3] Irina G, Alexey Z, Mikhail P, el al. Design and Implementation of a Block Storage Multi-protocol Converter[C]//Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies. San Diego, California, USA: [s. n.], 2003.

[4] 潘家铭, 舒继武, 张素琴, 等. Design and Implementation of SCSI Target Emulator[J]. Tsinghua Science and Technology, 2006, 11(2): 38-45.

[5] He Xubin. A Caching Strategy to Improve iSCSI Performance[C]//Proceedings of the 27th Annual IEEE Conference on Local Computer Networks. Tampa, Florida, USA: [s. n.], 2002.