AN ISCSI DESIGN AND IMPLEMENTATION

Hui Xiong, Renuga Kanagavelu, Yaolong Zhu, and Khai Leong Yong
Data Storage Institute,
DSI Building, 5 Engineering Drive 1,
Singapore 117608
Tel:+65-68748100, Fax: +65-68732745
{Xiong_Hui, Renuga_KANAGAVELU, ZHU_Yaolong, YONG_Khai_Leong}
@dsi.a_star.edu.sg

Abstract
iSCSI is a network storage technology designed to provide an economical solution over a TCP/IP Network. This paper presents a new iSCSI design with multiple TCP/IP connections. The prototype is developed and experiments are conducted for performance evaluation on Gigabit Ethernet (GE). Test results show that the new iSCSI design improves performance 20%~60% compared with normal iSCSI architecture. Throughput can reach 107MB/s for big I/O and I/O rate can reach 15000 IOPS for small I/O.

1. Introduction

iSCSI is a network storage technology which transports SCSI commands over TCP/IP network. It has attracted a lot of attention due to the following advantages:

a. Good scalability. iSCSI is based on SCSI protocol and TCP/IP network, which can provide good scalability.
b. Low-cost. iSCSI can share and be compatible with existing TCP/IP networks. The user does not need to add any new hardware.
c. Remote data transferring capability. TCP/IP network can extend to metro area, which makes iSCSI suitable for remote backup and disaster recovery applications.

Most current iSCSI implementations use software solutions in which iSCSI device drivers are added on top of the TCP/IP layer for off-the-shelf network interface cards (NICs). It may cause performance issue. Many iSCSI researches and projects have been carried out to analyze the problem. A prior research project of iSCSI – Netstation project of USC showed that it was possible for iSCSI to achieve the 80% performance of direct-attached SCSI device [1]. IBM Haifa Research Lab carried out research on the design and the performance analysis of iSCSI [2, 3]. Bell Laboratories also did some test and performance study of iSCSI over metro network [4]. Some of solutions have also been brought forward to handle the performance issue. A research group proposed a solution to use memory of iSCSI initiator to cache iSCSI data [5]. Other solutions included using a TCP/IP offload Engine (TOE) [6] and even iSCSI adapter [7] to reduce

the burden of host CPU by offloading the processing of TCP/IP and iSCSI protocol into the hardware on the network adapter. But these hardware solutions will add the extra cost compared to a software solution.

The improvement of semiconductor technology has led to the rapid increase of CPU speed and memory access speed. During the past two years, commercial CPU speed has almost tripled from 1GHz to 3GHz, and memory bandwidth has also doubled from around 200~300MB/s to around 400~500MB/s. A powerful hardware platform makes it possible to achieve good performance for iSCSI software solutions.

The paper presents a new software iSCSI design and implementation, which employs multiple TCP/IP connections. Experiments are conducted on the GE network both in the lab and metro network environment. Testing results are analyzed and discussed.

2. Software iSCSI Design

2.1 iSCSI Storage Architecture

Figure 1 shows an iSCSI storage architecture including an initiator and a target, which communicate to each other via the iSCSI protocol. In the initiator, the application, which needs to store and access data to/from the storage device, issues file requests. The file system converts file requests to block requests from application to block device layer and SCSI layer. The initiator iSCSI driver encapsulates SCSI commands in iSCSI Protocol Data Units (PDUs) and sends them to the Ethernet network via the TCP/IP layer. The target iSCSI driver receives iSCSI PDUs from TCP/IP layer and de-capsulates it. Then SCSI commands are mapped to RAM, called RAM I/O, or mapped to an actual magnetic storage disk, called DISK I/O. The target driver then sends response data and status back to the TCP/IP layer. The low-level flow control of iSCSI fully follows the

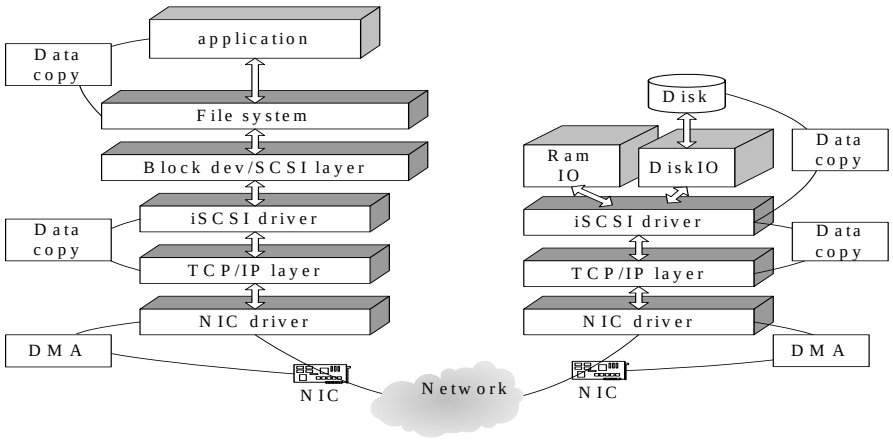

Fig.1 General iSCSI architecture

TCP/IP and Ethernet communication mechanism, which connects initiator and target by IP network. It is noted that two data copies and one DMA are processed in the initiator side during one IO access. For RAM I/O, one data copy and one DMA are processed in target.

2.2 A new iSCSI Design and Implementation

Normal iSCSI implementation is based on the single TCP/IP connection, which may not sufficiently utilize the network bandwidth. Furthermore, it may face serious performance issue caused by packets loss and long latency in metro network environment. We propose a new iSCSI architecture with multiple TCP/IP connections. This new architecture actually employs multiple virtual connections over one physical Ethernet connection (by one NIC), which is different from general idea of multiple Ethernet physical connections (by multiple NICs) according to iSCSI Request for Commands (iSCSI RFC) documents. The new design supposes not only to improve the iSCSI performance by increasing the utilization of network bandwidth, but also to provide a better mechanism to handle the long latency issue in metro network environment.

The working principle of our new design is shown in Figure 2. Multiple virtual TCP/IP connections are built on one physical Ethernet connection. One connection is used for sending SCSI request from initiator to target; another one is used for sending response from target to initiator. One pair of transmitting thread (Tx_thread) and receiving thread (Rx_thread), which locate in initiator or target respectively, are responsible for data communication within one connection.

We use the read operation as an example to explain the detailed communication procedure. The SCSI middle layer is a standard interface for SCSI layer to communicate with iSCSI device driver. The two main functions of SCSI middle layer are queuecommand( ) which issues SCSI command to the iSCSI driver and done( ) which informs SCSI middle layer that the command is finished by iSCSI driver. The iSCSI

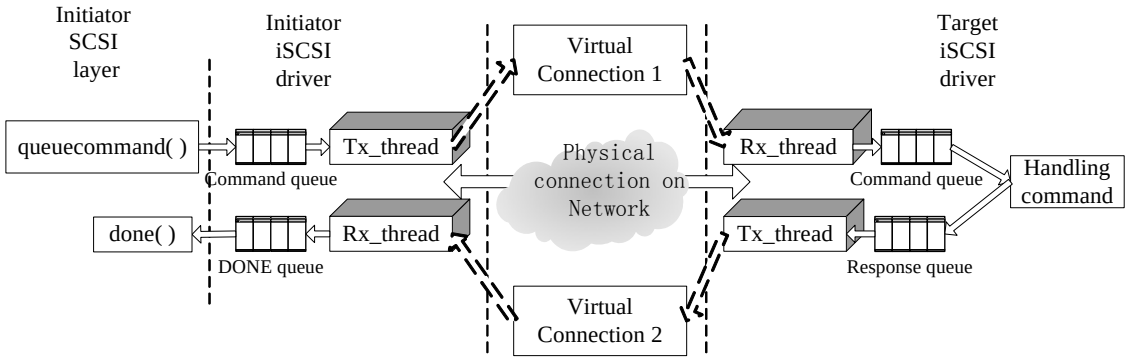

Fig 2. Multiple parallel connections iSCSI architecture

initiator driver gets read commands from SCSI middle layer. These commands are encapsulated in iSCSI request PDUs and then queued in the initiator command-queue. As shown in Figure 2, The Tx_thread of connection 1 sends these PDUs from command-queue to the target. The target driver receives these PDUs by the Rx_thread of the same connection. After de-encapsulating iSCSI request PDUs and mapping SCSI commands to the storage device, the target driver gets data from storage device and forms iSCSI response PDUs (including data and status). Then these iSCSI response PDUs are queued in the response-queue. Tx_thread in the target sends these PDUs by connection 2. The initiator driver receives response through the Rx_thread of the connection 2. Then the initiator driver put it to done-queue to call the done( ) function to finish the SCSI exchange.

As multiple connections are used, synchronization becomes important. According to iSCSI draft, the “initiator task tag” is used to record the sequence number of the iSCSI command in every iSCSI PDU. Related data and Status PDUs of the iSCSI command attach the same “initiator task tag”. Even if multiple connections are used and command queue is enabled in both initiator and target, the device driver can easily find respective data/status PDUs from the pending queue.

3. Experiments

Figure 3 shows the system configuration used in our iSCSI experiments in the lab environment. One Nera-summit-5i Gigabit Ethernet (GE) switch (supporting 9K jumbo frames) is used to connect iSCSI initiator and target. One Finisar GTX THG iSCSI analyzer is used to monitor and capture all Ethernet packets for detailed analysis. Further experiments are conducted in the metro network environment. iSCSI initiator and target are connected through fiber with the aid of DWDM switch over 25KM physical distance, as shown in Figure 4.

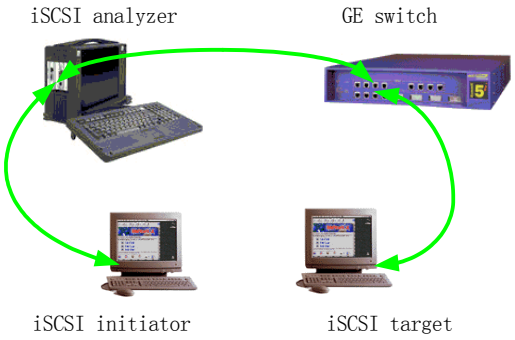

Fig. 3 Experiment platform in Lab

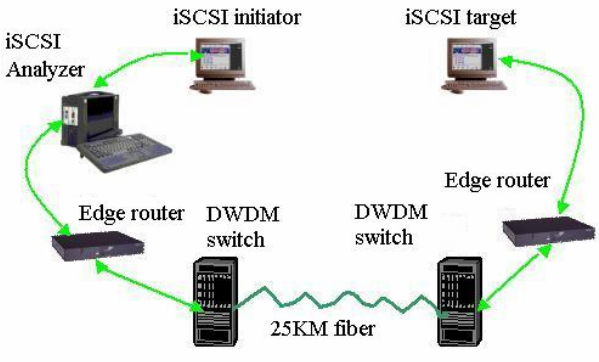

Fig.4 Metro network experiment

The hardware configuration of the iSCSI target includes a 64-bit/133MHz PCI-X motherboard with 2.4GHz P4 processor and 512M RAM. The iSCSI initiator is implemented on a 64-bit/66MHz PCI motherboard with 2.4GHz P4 processor and 256MB RAM. Intel PRO1000F GE network interface cards (NICs) are used at both iSCSI initiator and target. All experiments are based on Redhat Linux 8.0 (kernel version 2.4.20). TCP network performance of the system is tested by netperf. Result shows that throughput can reach 939.8Mbps with maximum 70% CPU utilization.

RAM I/O mode or DISK I/O mode is used in iSCSI target. RAM I/O directly maps requests to memory. In DISK I/O test, to diminish the impact from low speed storage device, we use a 2G fibre channel HBA to connect a self-developed high speed fibre channel disk array, which can reach around 110 MB/s for sequential write and 80 MB/s for sequential read.

We use dd command as the application benchmark tool to copy data to/from iSCSI disk. For example, “dd write” is “dd if=/dev/zero of=/dev/sda bs=4k count=500000”. It will generate write request to raw device. The size of request is set much bigger than initiator’s memory to avoid impact of local cache. In big I/O test, the general dd commands will generate SCSI commands, which request 128KB data to SCSI layer. In small I/O test, the kernel source code is modified to allow dd command to generate 1KB to 32KB requests to SCSI layer. Different queue lengths are tested for the small I/O test.

4. Experimental Results and Analysis

The iSCSI prototype with multiple connections has been tested and the results have been compared with the normal iSCSI performance. Figure 5 shows the throughput of iSCSI with RAM I/O mode .The I/O request size is 128KB. Ethernet frame size is set as 1.5k (small frame) or 9k (jumbo frame) respectively. The multiple connections iSCSI prototype can achieve 70~80MB/s for small frame test and 97~107MB/s for jumbo frame test, which is around 20%~60% improvement compared with normal iSCSI prototype. Further analysis of CPU utilization shows that utilization of initiator’s CPU can reach almost 100% with 107MB/s write throughput. That means CPU’s power become system’s bottleneck in this test condition.

The impact of the frame size on the iSCSI performance is shown in Figure 6. The bigger frame will improve performance mainly because it decrease interrupt times. The impact is very small when the frame size is bigger than 3k. CPU utilization in the initiator is always higher than target. This is because the initiator needs to handle one more data copy in memory than target, when RAM I/O is employed in target.

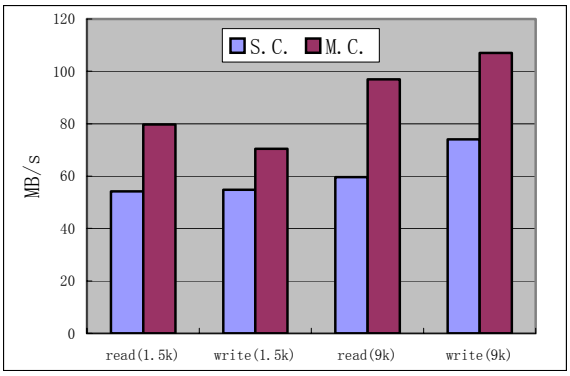

Fig. 5 single /multiple connection of RAM I/O

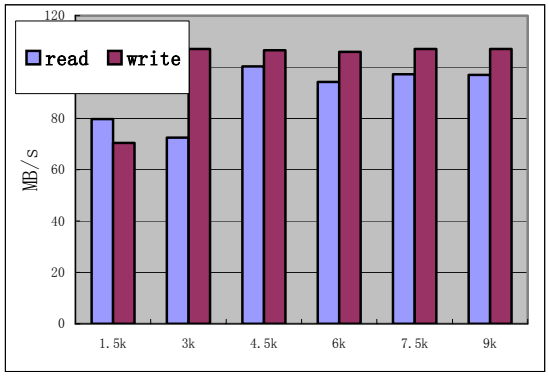

Fig. 6. Effects of Ethernet frame on the throughput

The performance of multiple connection iSCSI for small I/O request is tested. Since the queue depth is a critical parameter that affects the iSCSI performance in small I/O, the effects of the queue length on iSCSI performance is summarized in Figure 7. The testing condition is set as request size with 1KB and frame size with 1.5kB. When queue depth equals to 2, I/O rate is only about 4000 IOPS. When the queue length is bigger than 8, I/O rate can reach around 15000 IOPS. The results show that the I/O rate increases with the queue length until the queue length equals to 8. Further analyzing the captured data by iSCSI analyzer, we find that the maximum effective command queue length is 8 in our prototype and experiments.

Figure 8 shows test result of DISK I/O in the lab and metro network environment. In the lab environment, read performance of iSCSI working in DISK I/O mode is much lower than that in RAM I/O mode. This is because of one more data copy to hard disk in the iSCSI target. For write operation, write cache of raid array card in the iSCSI target makes the iSCSI write performance in DISK I/O mode almost same as that in RAM I/O mode.

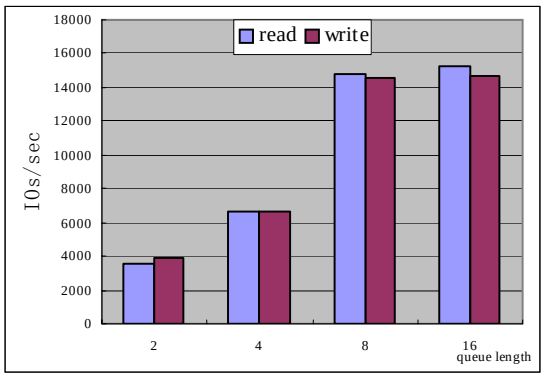

Fig.7 I/O rate for small I/O on different queue length

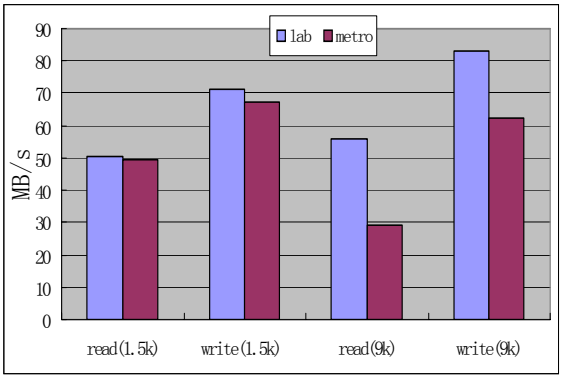

Fig.8 result of lab and metro network of DISK I/O

It is commonly supposed that network latency has a big impact on the iSCSI performance in a metro network. Network latency is estimated by the ping command. The roundtrip time is around 0.6ms in metro network which is around 20 times of that in lab network. But the throughput of our new iSCSI prototype can still reach 67MB/s with small frame in the experiment, which is only 2%~5% less than that of lab. This is due to the iSCSI queue architecture and multiple connections design. It seems that the metro network can't support jumbo frame well because it makes performance much worse than small frames.

5. Conclusion

The paper presents a new iSCSI design with multiple TCP/IP connections. The prototype has been developed and tested in both lab and metro network environment. Results show that the new iSCSI prototype can achieve 20%~60% performance improvement compared with normal iSCSI architecture with single TCP/IP connection. The new iSCSI architecture has been proved in the metro network environment. Future work will focus on the iSCSI testing and application in real network environment.

Reference

[1] Rodney Van Meter, Gregory G, Finn, Steve Hotz, "VISA: Netstation's Virtual Internet SCSI Adapter",
Proceedings of the eighth international conference on Architectural support for programming languages and operating systems, 1998

[2] Prasenjit Sarkar , Kaladhar Voruganti , "IP Storage: The Challenge Ahead"
Proc of 10th conference on Mass Storage Systems and Technologies,April-2002.

[3] Kalman Z.Meth, "iSCSI Initiator Design and Implementation Experience",
Proc of 10th conference on Mass Storage Systems and Technologies, April-2002.

[4] Wee Tech Ng, Bruce Hillyer, Elizabeth Shriver, Eran Gabber, Banu Özden, "Obtaining High Performance for Storage Outsourcing"
Proceedings of the Conference on File and Storage Technologies, 2002

[5] Xubin He, Qin Yang, Ming Zhang, " A Caching Strategy to improve iSCSI Performance",
Proc of the 27th Annual IEEE conference on Local Computer Networks (LCN'02), 2002.

[6] Alacritech, White paper, "Delivering High Performance Storage Networking",
http://www.alacritech.com/html/storagewhitepaper.html.

[7] Adaptec, White paper, "Building SANs with iSCSI, Ethernet and Adaptec",
http://www.graphics.adaptec.com/pdfs/buildingsanwithiscsi-21.pdf