Performance Evaluation of Block Device Layer with Automatic Parallelism Tuning with Heterogeneous IP-SAN Protocols

Takamichi Nishijima †, Hiroyuki Ohsaki †, Yoshihiro Nomoto ‡ and Makoto Imase †
† Graduate School of Information Science and Technology, Osaka University, Japan
{t-nisijm,oosaki,imase} @ist.osaka-u.ac.jp
‡ NTT Service Integration Laboratories NTT Corporation, Japan
nomoto.yoshihiro@lab.ntt.co.jp

Abstract

In this paper, we evaluate BDL-APT (Block Device Layer with Automatic Parallelism Tuning) *that maximizes the throughput of IP-SAN protocols in long-fat networks. BDL-APT parallelizes data transfer using multiple IP-SAN sessions at a* block device layer*, and adjusts the number of active IP-SAN sessions automatically according to network status. We perform quantitatively investigation on the effectiveness of BDL-APT in realistic network environments using our BDL-APT implementation with heterogeneous IP-SAN protocols (i.e., iSCSI, NBD, and GNBD). Consequently, we demonstrate that our BDL-APT operates effectively in long-fat networks and that the performance with NBD and GNBD protoocls can be further improved with parameter tuning.*

1 Introduction

In recent years, IP-SANs (IP-based Storage Area Networks) have been attracting attention for building SANs on IP networks [12,14]. Because of rapid advancement and development of networking technologies, strong requirements on remote backup using communication network has been emerging. One of major technologies for remote backup is SAN (Storage Area Network), which builds a network of storage devices connected by a communication network. IP-SANs have been widely used because of its low cost for building SANs and high compatibility with existing network infrastructures.

Several IP-SAN protocols such as iSCSI (Internet Small Computer System Interface) [13], NBD (Network Block Device) [8], GNBD (Global Network Block Device) [2] and iFCP (Internet Fibre Channel Protocol) [9] have been widely utilized and deployed for building SANs on IP networks. IP-SAN protocols allow interconnection of a remote diskvia a TCP/IP ewtwork. In this approach, a remote disk exports a portion of its storage space to a client. The client handles the remote disk no differently than its local disks – it runs a local file system that reads and writes data blocks to the remote disk.

IP-SAN protocols realize connectivity to remote storage devices over a conventional TCP/IP network, but they still have several issues to be solved — in particular, performance issues [7, 10]. IP-SAN protocols generally utilize TCP (Transmission Control Protocol) for data delivery, which results in low performance in a long-fat network. There exist several factors affecting the performance of IP-SAN protocols in a long-fat network. One of the most significant ones is the TCP performance degradation in a long-fat network [5].

Several solutions for improving the performance of IP-SAN protocols have been proposed [3, 5, 15]. For instance, to prevent throughput degradation of iSCSI protocol, solutions utilizing multiple links [15] or parallel TCP connections [3] have been proposed. In [15], iSCSI throughput is improved using multiple connections, each of which traverses a different path using multiple LAN ports and dedicated routers. Also, iSCSI throughput is improved by adjusting the number of parallel TCP connections using the parallel data transfer feature of iSCSI protocol [3]. On the contrary, a changes to the TCP congestion control algorithm for improving fairness and throughput of iSCSI protocol have been proposed [5].

In our previous work [11], we have proposed *BDL-APT (Block Device Layer with Automatic Parallelism Tuning)*, which maximizes the performance of heterogeneous IP-SAN protocols in a long-fat network. BDL-APT parallelizes data transfer using multiple IP-SAN sessions at a *block device layer*, and adjusts the number of active IP-SAN sessions automatically according to network status. A block device layer is a layer that receives read/write requests from an application or a file system, and relays those requests to

a storage device. BDL-APT parallelizes data transfer by dividing aggregated read/write requests into multiple chunks, and then transfering a chunk of requests on every IP-SAN session in parallel. BDL-APT automatically optimizes the number of IP-SAN sessions based on the measured network status using our APT mechanism [3,4].

In [11], the effectiveness of BDL-APT with NBD protocol has been demonstrated through preliminarily experiments. We have performed preliminary investigation on the effectiveness of our BDL-APT using our BDL-APT implementation and network emulator. We have demonstrated that our BDL-APT operates effectively in long-fat networks.

In this paper, we evaluate the performance of BDL-APT with heterogeneous IP-SAN protocols (i.e., iSCSI, NBD, and GNBD) in a long-fat network. We implemented BDL-APT as a layer of MD (Multiple Device) driver, which is one of major software RAID implementations included in Linux kernel. We perform quantitatively investigation on the effectiveness of our BDL-APT in realistic network environments using our BDL-APT implementation.

Through extensive experiments, we show that the effectiveness of BDL-APT with heterogeneous IP-SAN protocols, and that the performance with NBD and GNBD protoocls can be further improved with parameter tuning. Consequently, we demonstrate that our BDL-APT operates effectively in long-fat networks regardless of IP-SAN protocols. The result show that the performance bottleneck of BDL-APT is TCP window and incressding the window size improves the throughput even with NBD, GNBD.

The organization of this paper is as follows. Section 2 summarizes heterogeneous IP-SAN protocols. Our BDL-APT is described in Section 3. Section 4 is devoted for performance evaluation of our BDL-APT using our BDL-APT implementation. Finally, Section 5 summarizes this paper and discusses future works.

2 IP-SAN Protocols

2.1 iSCSI (Internet Small Computer System Interface)

The iSCSI (Internet Small Computer System Interface) protocol encapsulates a stream of SCSI CDBs (Command Descriptor Blocks) in IP packets, and it was standardized by IETF in 2004 [13]. iSCSI simply allows interconnection of SCSI devices via a TCP/IP network. When SCSI device receives read/write requests from an application or a file system, SCSI device generates SCSI CDBs and transfers these SCSI CDBs to local SCSI storage. iSCSI initiator provides SCSI devices operating at users side i.e., acts as clients, and iSCSI target operates at storage side and perform storage access. When iSCSI initiator reveives read/write requests, iSCSI initiator generates SCSI CDBs and transfers these SCSI CDBs to iSCSI target using TCP connections. When iSCSI target receives SCSI CDBs from iSCSI initiator, iSCSI target transfers these SCSI CDBs to local SCSI strage. Hence, using the iSCSI protocol, applications and file systems can extend their reachability to remote storage devices as well as local ones.

It has been known that iSCSI performance is significantly degraded when the end-to-end delay (i.e., the delay between iSCSI initiator and target) is large [1, 3, 6, 7]. There have been several researches on performance evaluation of the iSCSI protocol in long-fat networks [1, 6, 7]. In [1, 6, 7], the performance of the iSCSI protocol is evaluated using experiment [6], simulation [7] or mathematical analysis [1]. Consequently, it has been shown that the iSCSI throughput is significantly degraded when the end-to-end delay (i.e., the delay between iSCSI initiator and target) is large. In [3], iSCSI throughput is improved by adjusting the number of parallel TCP connections using the parallel data transfer feature of iSCSI protocol regardless of link delay.

2.2 NBD (Network Block Device)

The NBD (Network Block Device) protocol is a light-weight IP-SAN protocol for accessing a remote block device through a TCP/IP network, and it was initially developped by Pavel Machek in 1997 [8]. NBD simply allows interconnection of Block devices via a TCP/IP network. NBD offers an access model that simulates a block device, such as a locale strage, on the locale client, but connects across the network to a remote server that provides the real physical storage. Even though the actual access requests and data blocks are communicated on the network, NBD layer hides all the details and the client simply uses the virtual devices as if it were a local block device.

It has been known that the NBD performance is sensitive to the end-to-end delay [11]. There have been a research on performance evaluation of the NBD protocol in long-fat networks [11]. In [11], the performance of the NBD protocol is evaluated using experiment. Consequently, it has been shown that the NBD goodput is improved by the adjustment of the number of parallel TCP connections.

2.3 GNBD (Global Network Block Device)

The GNBD (Global Network Block Device) protocol is another light-weight IP-SAN protocol for accessing a remote block device through a TCP/IP network, and it was developped in the University of Minnesota as a part of GFS (Global File System) [2]. GNBD simply allows interconnection of Block devices via a TCP/IP network. GNBD adds the functionality of resource exclusion control to NBD intending for the use especially in clusters. The large differ-

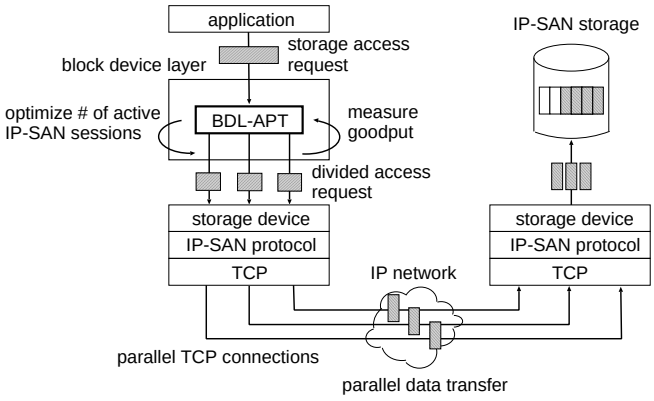

Figure 1: BDL-APT overview; BDL-APT parallelizes data transfer using multiple IP-SAN sessions at a block device layer, and adjusts the number of active IP-SAN sessions automatically according to network status.

ence between NBD and GNBD is that GNBD alllows multilpe clients to across the same block device concurrently while NBD driver only allows a single client at a time. Today, it is maintained by Red Hat. Inc. and availabe to everyone under an open source lisence.

3 BDL-APT (Block Device Layer with Automatic Parallelism Tuning)

3.1 Overview

BDL-APT (Block Device Layer with Automatic Parallelism Tuning) automatically maximizes the performance of heterogeneous IP-SAN protocols in long-fat networks. BDL-APT parallelizes data transfer using multiple IP-SAN sessions at a block device layer, and adjusts the number of active IP-SAN sessions automatically according to network status. BDL-APT is a mechanism that operates as a block device layer at an IP-SAN initiator (i.e., client) (see Fig. 1). BDL-APT parallelizes data transfer by dividing aggregated read/write requests into multiple chunks, and then transfering a chunk of requests on every IP-SAN session in parallel. BDL-APT automatically optimizes the number of active IP-SAN sessions based on the measured network status using our parallelism tuning mechanism APT [4], which is based on a numerical computation algorithm called Golden Section Search method.

3.2 Building blocks

BDL-APT is composed of three main building blocks for parallelizing data transfer, for monitoring data transfer throughput and delay, and for optimizing the number of multiple IP-SAN sessions.

- Parallelizing data transfer

  BDL-APT realizes parallel data transfer by establishing multiple IP-SAN sessions to a single storage. When BDL-APT receives read/write requests (it is hereafter called *block I/O request*) from an application or a file system, BDL-APT splits those block I/O requests into multiple chunks. BDL-APT generates multiple block I/O requests for each chunk. BDL-APT parallelizes data transfer by assigning those generated block I/O requests to multiple IP-SAN sessions.

- Optimizing the number of multiple IP-SAN sessions

  BDL-APT adjusts the number of parallel TCP connections by adjusting the number of active IP-SAN sessions. An IP-SAN protocol utilizing TCP for data delivery establishes at least one TCP connection per an IP-SAN session. Thus, it is possible to adjust the number of parallel TCP connections by adjusting the number of active IP-SAN sessions. BDL-APT maintains multiple IP-SAN sessions. BDL-APT determines the required number of parallel TCP connections, and adjusts the number of active IP-SAN sessions. By assigning generated block I/O requests to a subset of established IP-SAN sessions, BDL-APT adjusts the number of active IP-SAN sessions used for parallel data transfer. BDL-APT determines the required number of IP-SAN sessions using our parallelism tuning APT mechanism [4].

- Monitoring data transfer goodput

  BDL-APT measures the goodput for every chunk transfer at a block device layer during parallel data transfer. When BDL-APT assigns generated block I/O requests to multiple IP-SAN sessions, BDL-APT records the size of each data transfer request. BDL-APT calculates the goodput of an IP-SAN protocol from the chunk size (i.e., the total size of block I/O requests consisting of a chunk) and the time required to transfer all block I/O requests in a chunk.

3.3 Implementation

We implemented BDL-APT in the MD (Multiple Device) driver, which is one of popular software RAID implementations included in the Linux kernel. The MD driver provides virtual devices that are created from one or more

independent underlying devices. This array of devices often contains redundancy, and hence the acronym RAID such as RAID0, RAID1 and RAID5. MD layer is a block device layer which is created above the layer of device drivers.

The BDL-APT module in the MD driver is derived from the RAID-0 module with adding several functions required for BDL-APT: parallelizing data transfer, optimizing the number of multiple IP-SAN sessions, and monitoring data transfer goodput. Parallelizing data transfer is implemented by just using the function of RAID0, but BDL-APT changes the number of active IP-SAN sessions according to network status. Optimizing the number of multiple IP-SAN sessions is implemented by using our parallelism tuning mechanism APT [3, 4]. Monitoring data transfer goodput is implemented by calculating from the total data size of Block I/O requests that transfer was completed at a uniform pace, which was a parameter of BDL-APT.

BDL-APT utilizes multiple IP-SAN sessions for improving the IP-SAN performance, so the maximum number of IP-SAN sessions in the MD driver and IP-SAN protocols (i.e., iSCSI, NBD, and GNBD) is increased. We modified the definition in header files of MD driver to increase the number of IP-SAN sessions which can be maintained. To establish multiple IP-SAN session using iSCSI, multiple iSCSI interfaces are generated. The number of the iSCSI session able to be established concurrently is configured at iSCSI target, we made it able to establish 128 sessions at most modifying configuration. We also increased the number of NBD sessions able to be established concurrently by modifying the header file of NBD protocol.

4 Experiment

4.1 Experiment design

We evaluate the performance of BDL-APT with heterogeneous IP-SAN protocols (i.e., iSCSI, NBD, and GNBD) in a long-fat network. To show the effectiveness of BDL-APT in realistic network, we conducted the experiment of our BDL-APT with iSCSI, NBD, and GNBD varying bandwidth and latency. It is known that the performance of both iSCSI and NBD drop in long-fat network. Though neither NBD and GNBD has the functionality of parallel TCP connections, we show that the performance can be improved by the adjustment of the number of parallel TCP connections using BDL-APT.

The network configuration used in our experiments composed of IP-SAN client and storage, and the network emulator (see Fig. 2). We transfered data from an IP-SAN client to an IP-SAN storage. We used the same computers with an Intel Xenon 3.06 [GHz] processor with 2 [GByte] memory for the IP-SAN client, IP-SAN target and the network emulator. For the IP-SAN client, we used Open iSCSI

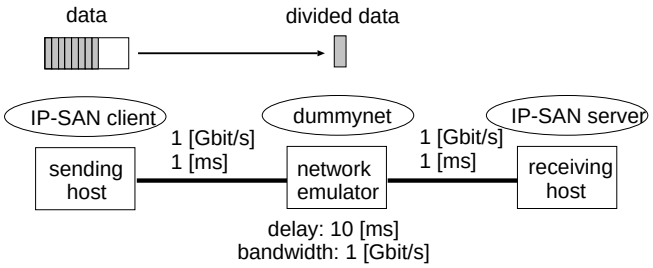

Figure 2: Network configuration userd in experiments: IP-SAN client and storage, and network emulator

Table 1. Parameter configuration used in experiments

Bandwidth of network emulator	1,000	[Mbit/s]
Delay of network emulator	10	[ms]
TCP socket buffer size	1024	[Kbyte]
Initial number of NBD sessions N_0 [4]	1	
Multiplicative increase factor α [4]	2	
Target value of chunk transfer time Δ [4]	10	[s]
Transfer time	300	[s]
Device size	512	[MB]

2.6-870, GNBD client 1.03.00 and NBD server 2.8.7 running on Debian GNU/Linux 3.1 (ethc)(Linux kernel 2.6.18). For the IP-SAN storage, we used the iSCSI target 0.4.17, GNBD server 2.03.09 and NBD server 2.9.11 running on Debian GNU/Linux 3.1 (lenny)(Linux kernel 2.6.26). We used FreeBSD 6.4 and dummynet for the network emulator. Unless explicitly stated, parameters shown in Tab. 1are used in the following experiments.

4.2 Effect of network bandwidth

First, the goodput of heterogeneous IP-SAN protocols (i.e., iSCSI, NBD, and GNBD) with BDL-APT in steady state is measured by changing the bottleneck link bandwidth (i.e., the bandwidth throttling at the network emulator) (see Fig. 3). We conducted three experiments and measured the average and 95% confidence interval of IP-SAN goodput. Figure 3 shows the IP-SAN goodput (i.e., aggregated IP-SAN goodput of all active IP-SAN sessions) when the bandwidth of the network emulator is changed as 100–1000 [Mbit/s].

One can find from Fig. 3 that BDL-APT fully utilizes the bottleneck link bandwidth regardless of IP-SAN protocols. Figure, 3 shows that BDL-APT can maximize the throughput of all of iSCSI, NBD, and GNBD regardress of band-

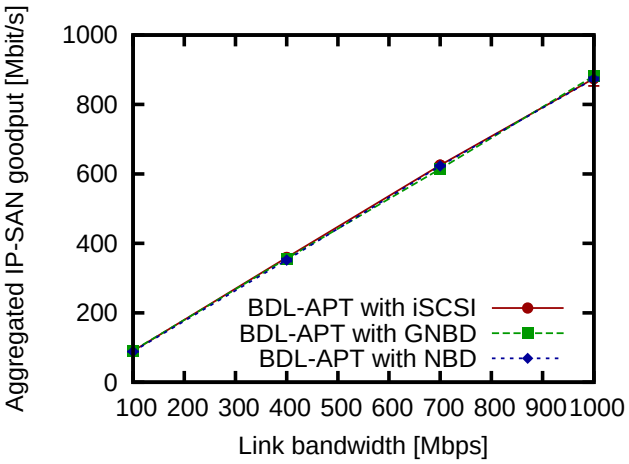

Figure 3: Bottleneck link bandwidth vs. goodput of IP-SAN protocols with BDL-APT; BDL-APT fully utilizes the bottleneck link bandwidth regardless of IP-SAN protocols.

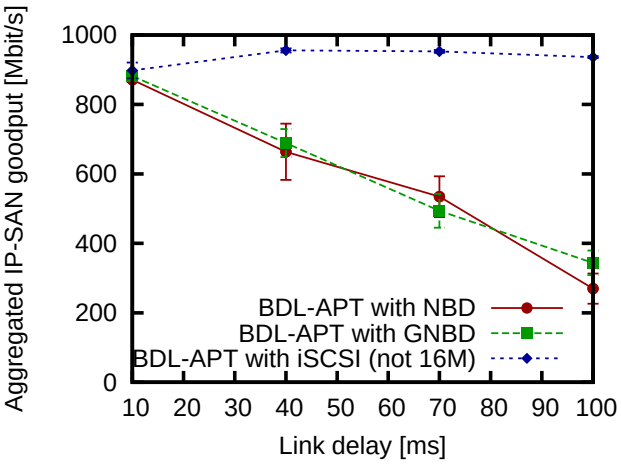

Figure 4: Bottleneck link delay vs. goodput of IP-SAN protocols with BDL-APT; BDL-APT maximizes iSCSI throughput regardless of latency and improves the throughput of GNBD outperforms that of NBD, but when the latency is larger than 40 [ms], the throughput slightly drops, which can be caused by inadequate size of TCP window.

width. Thus, it can be said that BDL-APT can fully utilize the bandwidth under the low latency environment no matter what the IP-SAN protocols.

4.3 Effect of network delay

Second, the goodput of heterogeneous IP-SAN protocols (i.e., iSCSI, NBD, and GNBD) with BDL-APT in steady state is measured by changing the network delay (i.e., the delay at the network emulator) (see Fig. 4). We conducted 10 experiments and measured the average and 95% confidence interval of IP-SAN goodput. Figure 4 shows the IP-SAN goodput when the delay of the network emulator is changed as 10–100 [ms].

One can find from Fig. 4, 5 that BDL-APT fully utilizes the bottleneck link delay regardless of IP-SAN protocols. Fig.4 shows that BDL-APT maximizes iSCSI throughput regardless of latency. On the other hand, the figure shows that the throughput of NBD and GNBD drops when the latency is high. Our further investigation identified TCP window as the performance bottoleneck of NBD and GNBD. Fig. 5 presents the result of BDL-APT's throughput improvement of NBD and GNBD under the TCP window-size of 16 [Mbyte]. NBD performs well regardless of latency, and the throughput of GNBD is maximized when the latency is under 40 [ms]. When the latency is under 40 [ms], the throughput of GNBD outperforms that of NBD, but when the latency is larger than 40 [ms], the throughput slightly drops, which can be caused by inadequate size of TCP window. These results show that BDL-APT can maximaze throughput regardless of the IP-SAN protocols and the bandwidth of bottoleneck link, however the size of TCP window must be adjusted for each IP-SAN protocols.

5 Conclusion

In this paper, we have evaluated BDL-APT (Block Device Layer with Automatic Parallelism Tuning) that maximizes the throughput of IP-SAN protocols in long-fat networks.

We have performed quantitatively investigation on the effectiveness of BDL-APT in realistic network environments using our BDL-APT implementation. Consequently, we have demonstrated that our BDL-APT operates effectively in long-fat networks. we have demonstrated that our BDL-APT operates effectively in long-fat networks regardless of IP-SAN protocols and that the performance with NBD and GNBD protoocls can be further improved with parameter tuning.

As future work,

we are planning to experiments on real network (SINET3) and

References

[1] C. M. Gauger, M. Kohn, S. Gunreben, D. Sass, and S. G. Perez. Modeling and performance evaluation of iSCSI stor-

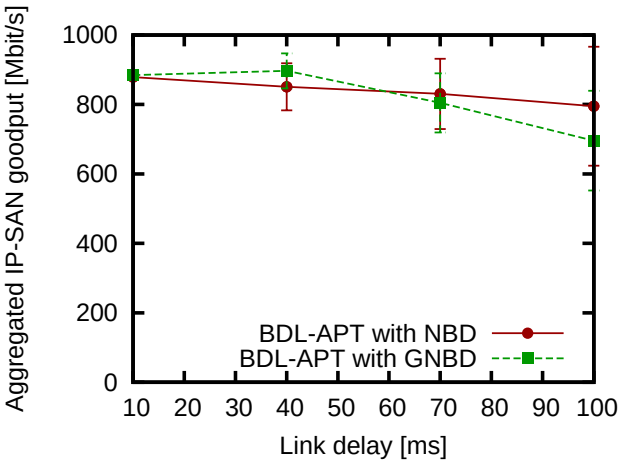

Figure 5: Bottleneck link delay vs. goodput of IP-SAN protocols with BDL-APT when TCP window size is 16 [Mbyte]; BDL-APT fully utilizes the bottleneck link bandwidth regardless of IP-SAN protocols when TCP window size is large.

age area networks over TCP/IP-based MAN and WAN networks. *Proceeding of the Second International conference on Broadband Networks(BROADNETS 2005)*, pages 915–923, Oct. 2005.

[2] GNBD Project Page. `http://sourceware.org/cluster/gnbd/`.

[3] F. Inoue, H. Ohsaki, Y. Nomoto, and M. Imase. On maximizing iSCSI throughput using multiple connections with automatic parallelism tuning. *in Proceedings of the 5th IEEE International Workshop on Storage Network Architecture and Parallel I/Os (SNAPI 2008)*, pages 11–16, Sept. 2008.

[4] T. Ito, H. Ohsaki, and M. Imase. GridFTP-APT: Automatic parallelism tuning mechanism for data transfer protocol GridFTP. In *Proceedings of 6th IEEE International Symposium on Cluster Computing and the Grid (CCGrid2006)*, pages 454–461, May 2006.

[5] B. K. Kancherla, G. M. Narayan, and K. Gopinath. Performance evaluation of multiple TCP connections in iSCSI. In *Proceedings of the 24th IEEE Conference on Mass Storage Systems and Technologies*, pages 239–244. IEEE Computer Society, Sept. 2007.

[6] Y. Lu and D. H. C. Du. Performance study of iSCSI-based storage subsystems. *IEEE Communications Magazine*, 41(8):76–82, Aug. 2003.

[7] Y. Lu, N. Farrukh, and D. H. C. Du. Simulation study of iSCSI-based storage system. In *Proceedings of 12th NASA Goddard & 21st IEEE Conference of Mass Storage Systems and Technologies (MSST 2004)*, pages 101–110, Apr. 2004.

[8] P. Machekz. Network Block Device. `http://nbd.sourceforge.net`.

[9] C. Monia et al. iFCP - a protocol for internet fibre channel storage networking. *Request for Comments (RFC) 4172*, Sept. 2005.

[10] W. Ng et al. Obtaining high performance for storage outsourcing. In *Proceedings of the 1st USENIX Conference on File and Storage Technologies*, pages 145–158, Jan. 2002.

[11] T. Nishijima, F. Inoue, H. Ohsaki, Y. Nomoto, and M. Imase. On maximizing IP-SAN throughput over TCP connections with automatic parallelism tuning for long-fat networks. In *Proceedings of the Third Workshop on Middleware Architecture in the Internet (MidArc 2009)*, pages 251–254, July 2009.

[12] P. Sarkar and K. Voruganti. IP storage: The challenge ahead. In *in Proceedings of the IEEE Symposium on Mass Storage Systems*, pages 35–42, 2002.

[13] J. Satran, K. Meth, C. Sapuntzakis, M. Chadalapaka, and E. Zeidner. Internet small computer systems interface (iSCSI). *Request for Comments (RFC) 3720*, Apr. 2004.

[14] P. Wang et al. IP SAN – from iSCSI to IP-addressable ethernet disks. *in Proceedings of the 20 th IEEE Conference on Mass storage Systems and Technologies*, page 189, Apr. 2003.

[15] Q. K. Yang. On performance of parallel iSCSI protocol for networked storage systems. In *Proceeding of the 20th International Conference on Advanced Information Networking and Applications (AINA 2006)*, volume 1, pages 629–636, Apr. 2006.