Optimization of iSCSI Remote Storage Access through Multiple Layers

Reika Higa[1] Kosuke Matsubara[2] Takao Okamawari[2]
Saneyasu Yamaguchi[3] Masato Oguchi[1]
[1]Graduate School of Humanities and Sciences, Ochanomizu University
2-1-1, Otsuka, Bunkyo-ku, Tokyo, 112-8610 Japan
[2]SOFTBANK TELECOM Corp
1–9–1, Higashishinbashi, Minatoku, Tokyo 105–7316, JAPAN
[3]Department of Computer Science and Communication Engineering, Kogakuin University
1-24-2, Nishishinjuku, Shinjuku, Tokyo 163-8677 Japan
E-mail: reika@ogl.is.ocha.ac.jp, oguchi@computer.org

Abstract

iSCSI has a problem of drastic performance deterioration in the case of longer-latency. Thus, we have optimized iSCSI remote storage access through multiple layers. As a result, when RTT is 32ms, performance of the optimized iSCSI has achieved quadruple of that of the default iSCSI. Moreover, we have constructed the model of iSCSI sequential access and analyzed the factor of iSCSI performance deterioration.

1. Introduction

As the volume of data that computer systems process increases, efficient management of storage becomes important. Because iSCSI is able to construct the wide area IP-SAN with low cost and execute easy data backup to remote place, for example data center, iSCSI is expected to be used as Storage Outsourcing service. However, iSCSI has a problem of drastic performance deterioration in the case of longer-latency. This is because iSCSI has a complex hierarchical structure, SCSI over TCP/IP over Ethernet. For the purpose of getting the better performance of iSCSI, we should control not only iSCSI layer but also other layers.

Thus, in this paper, we have optimized iSCSI remote storage access through multiple layers. Moreover, we have constructed the model of iSCSI sequential access and analyzed the factor of iSCSI performance.

2. Background of our research works

2.1. iSCSI

iSCSI is a typical protocol of IP-SAN. iSCSI is a standard which encapsulates a SCSI command into a TCP/IP packet, and thus we can build SAN only with IP equipment using iSCSI.

On the other hand, it is complicated hierarchical structure as shown in Figure 1. Therefore it cannot exceed the limit performance of its lower protocol. In addition, although iSCSI is expected to be used for the long-distance access, it has the problem of Long Fat Pipe when it uses a high-speed connection like a gigabit network. Taking these matters into account, an appropriate control through multiple protocol layers is required for the optimization of iSCSI remote storage access.

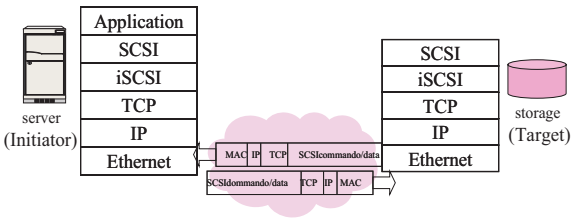

Figure 1. Configuration of iSCSI

2.2. TCP Congestion Window control algorithm

TCP uses the concept of Congestion Window (CWND) for congestion control of a network. CWND indicates the number of maximum packets that can be sent consecutively without receiving a reply packet of Acknowledgement (ACK) from a data receiver. That is to say, CWND is a parameter, which limits the behavior of a data sender, for the purpose of the network congestion control. Generally, CWND increases whenever a data sender receives one ACK, if the state of communications of TCP is judged as normal. However, if TCP implementation detects an error and judges the state of communication as abnormal, CWND

reduces dramatically. In Linux TCP implementation, the cases in which CWND reduces are as follows (Figure 2).

1. CWR: Detecting Local Congestion error in which device driver buffer of the data sender overflows.

2. Recovery: Receiving duplicated ACKs or SACK.

3. Loss: Detecting timeout.

Linux TCP Implementation does not increase the window size unless CWND is consumed completely before receiving ACKs. In such a case, we confirmed the throughput remains stable.

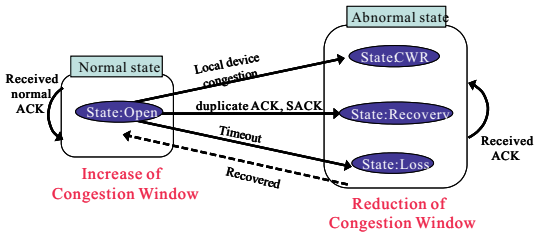

Figure 2. State transition of LinuxTCP implementaion

There are various sorts of Congenstion Window Control Algorithm implemented in Linux kernel version 2.6. We can change the algorithm easily by the user commands. In our research work, four sorts of algorithm are used for the evaluation. They are Reno, Binary Increase Congestion Control (BIC), Westwood, and Hamilton TCP (H-TCP) [1].

Reno is the basic algorithm. A wide variety of algorithm has been developed based on Reno. Reno detects congestion by packet-loss and it regards transfer rate at the time as available bandwidth. For example, if three consecutive Duplicate Acks are received, Reno regards it as occurring of packet-loss and reduces CWND by half. And it increases CWND on receiving every ACK. Thus, Reno is algorithm that increases the size of CWND gradually and drops it by detecting congestion.

BIC is default algorithm in OS of our experimental environment. In normal TCP congestion control, linear search is executed for available bandwidth. On the other hand, binary search is executed in BIC.

Westwood is optimized for lossy networks. Westwood is the algorithm optimized based on Reno for a wireless environment.

H-TCP is recommended for high-speed and long-distance networks. It is designed to recover quickly after congestion.

2.3. Existing research works

As a research work on iSCSI, the performance of original SCSI over IP is measured and analyzed in [2]. In [3], the comparison is done between iSCSI software implementation and hardware implementation which is realized using TCP Offload Engine(TOE) or Host Bus Adapter(HBA). Although a load of CPU can be reduced in the case of hardware, iSCSI software implementation is better for overall performance. In [4], if the measured and analyzed performance of iSCSI and FC-SAN. As a research on transport layer which is changed from TCP Reno to other TCP, the simulation result is reported in [5]. In [6], the achievement of performance upgrade is reported by reconfiguring iSCSI implementation. In [7], the influence of iSCSI protocol parameters to performance is discussed thoroughly. In [8], the result of applying Fair-TCP, in which TCP control information is shared among multiple TCP connections, to the case of multiple iSCSI connections is reported. As a research work on the performance of TCP in a long distance network, Data Reservoir Project is known[9].

3. Experimental system

3.1. TCP CWND monitor tool

Generally, user programs cannot recognize the size of CWND because CWND is a parameter controlled in a Kernel space of an operating system. Therefore we have inserted monitor functions in TCP source code and implemented a recording mechanism of TCP parameters within a Linux Kernel memory space, so that they are accessible from User space as shown in figure 3. With this mechanism, we can confirm TCP parameters by reading a special file for accessing Kernel memory space. What we can monitor in this tool are the value of CWND and various error events (Local device congestion, duplicate ACK/SACK, and Timeout).

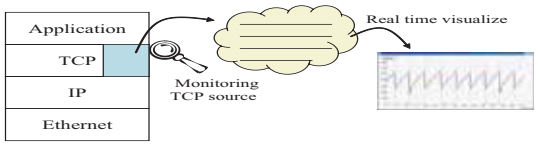

Figure 3. TCP CWND monitor tool

3.2. Experimental environment

In this experiment, Initiator and Target are connected by Gigabit Ethernet, and TCP/IP connection is established between them. As target storage, SAS disks are used with RAID0 configuration, which is a common way of building high-performance storage in these days. Experimental environment and specification are shown in Figure 4 and Table 1.

Iperf [10] is used for the measurement of network throughput. Bonnie++ [11] is used for the evaluation of storage access.

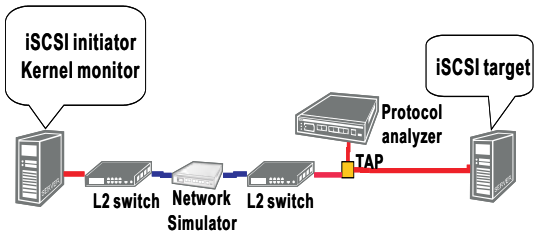

Figure 4. Experimental environment

Table 1. Experimental setup : PCs

OS	Red Hat Enterprise Linux 2.618-8.e.15
CPU	Quad Core Intel Xeon 1.6GHZ
Main Memory	2GB
NIC	Intel PRO/1000PT Server Adaptor on PCI Express
HDD	73GB SAS × 2(RAID0)
RAID Controller	SAS5/iR
iSCSI	Initiator : open-iscsi-2.0-865
	Target : iSCSI Enterprise Target(IET)-0.4.15
Network Analyzer	ClearSight Network Recorder
Network Simulator	ANUE

For achieving the better performance of iSCSI, we should control not only iSCSI layer but also other layers. In our experiment, we have optimized and evaluated SCSI/iSCSI layer, TCP/IP layer, and Ethernet layer. We have optimized iSCSI parameters in SCSI/iSCSI layer, evaluated the performance of iSCSI Storage Access in the case of changing Congestion Window control algorithm in TCP/IP layer, and optimized NIC parameters in Ethernet layer.

4. Optimization of iSCSI parameters

We have optimized iSCSI parameters for the better performance of write storage access. The optimized parameters are shown in Table 2.

Table 2. iSCSI parameters setup

	Default	Optimized
Target Side		
InitialR2T	Yes	No
ImmediateData	No	Yes
FirstBurstLength	65536	1048576
MaxBurstLength	262144	1048576
MaxRecvDataSegmentLength	8192	262144
Initiator Side		
node.conn[0].iscsi.MaxRecvDataSegmentLengt	131072	262144
node.session.iscsi.FirstBurstLength	262144	1048576

First, a long-latency environment is configured by using a network simulator, in which RTT is 0-32ms. Next, we have measured throughput of the default iSCSI and optimized iSCSI storage access using “ dd ” command. In this access, the block size is 10MB and the number of access is 100. For comparison, we have measured the performance of a local disk (SAS) access by using Bonnie++. They are shown Figure 5. According to this graph, performance of local access is extremely high with the effect of RAID0 configured SAS disks. On the other hand, performance of iSCSI accesses is lower than that of local access. However, in the case of low-latency, performance of iSCSI access is relatively good. Moreover, performance of optimized iSCSI is definitely better than that of default iSCSI. For example, when RTT is 32ms, performance of the optimized iSCSI is quadruple of that of the default iSCSI. However, even in such a case, the optimized iSCSI comes to be drastic performance deterioration as it becomes longer-latency.

In Figure 5, throughput of socket access is also shown for comparison with iSCSI access. In socket access, 16MB of the advertisement window is used as enough size of communication, and long-latency is set by using a network simulator, in which RTT is 0-32ms. The measurement time is 1000s. According to Figure 5, performance of socket access keeps good throughput even in the case of long-latency. Therefore, the goal of the optimization is to prevent drastic performance deterioration and keep good performance even in iSCSI access with long-latency. It is known that there are relationship between throughput and the size of TCP CWND. Thus, we consider that Congestion Window control algorithm is related to the cause of the decrease of throughput. The experiment of changing Congestion Window control algorithm in TCP/IP is described in the next section.

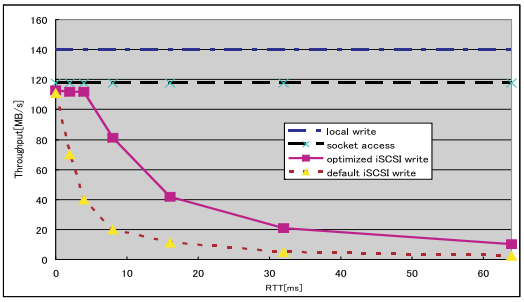

Figure 5. Comparison of throughput of default iSCSI, optimized iSCSI, and socket access

5. Performance evaluation in the case of changing Congestion Window control algorithm in TCP/IP

We have tried to evaluate the effectiveness of changing Congestion Window control algorithm in TCP/IP with iSCSI access. First, we have evaluated the socket access case as a basic experiment. Next, the case of iSCSI access has been evaluated.

5.1. Evaluation of performance in socket access

We have used four sorts of algorithm as Congestion Window control algorithm. There are Reno, BIC, Westwood, and H-TCP. We have measured throughput of socket access with each algorithm by Iperf. The measurement is executed five times respectively and the result is calculated by the average value of three except Maximum and Minimum. RTT is 8-64ms. The measurement length is 1000s. Figure 6 shows the throughput of the socket access. In Figure 6, BIC is the same as socket access in Figure 5.

In order to clarify the difference of each algorithm, throughput of 30s from the beginning has also measured. Figure 7 shows the result. Figure 6 shows that the difference of each algorithm is confirmed with the longer latency. Because the difference is larger in Figure 7 than that in Figure 6, there should be a cause of difference at the beginning of communications. In the case of long latency, the behavior of each algorithm is greatly different especially at the beginning. As a result, there should be a difference of throughput of each algorithm as a whole.

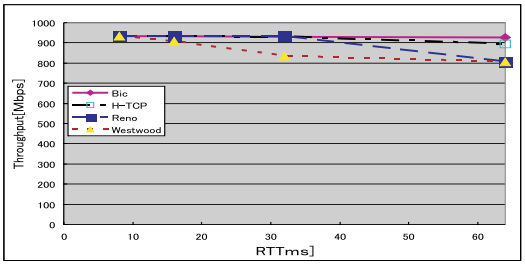

Figure 6. Comparison of throughput of each algorithm (1000s)

5.2. Evaluation of performance in iSCSI access

We have measured throughput of iSCSI access with each algorithm by " dd " command. The block size is 10MB and the number of total accesses is 100. The measurement is executed five times respectively and the result is calculated by the average value of three except Maximum and Minimum values. RTT is 8-64ms. In iSCSI access, the difference of throughput, when the Congestion Window control algorithm is changed, has not been confirmed, different from the case of socket access. The result of all algorithms is the same, which is shown in Table3.

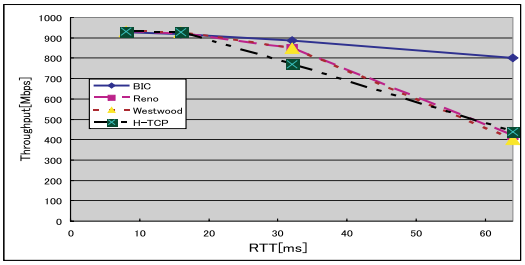

Figure 7. Comparison of throughput of each algorithm (30s)

Table 3. iSCSI throughput in long-latency

RTT	8ms	16ms	32ms	64ms
Throughput	80MB/s	46MB/s	24MB/s	12MB/s

5.3. Discussion

In iSCSI accsess, the difference of throughput, which has been observed in sockets access, is not confirmed when Congestion Window control algorithm in TCP/IP is changed. In our experimental environment, the change of Congestion Window control algorithm in TCP/IP does not influence performance of iSCSI. Thus, the performance improvement should be absorbed in the sequence of iSCSI block access and has disappeared.

6. Optimization of NIC parameters

For further optimization of iSCSI hierarchy protocol, we have optimized NIC parameters in Ethernet layer. We have compared the optimized performance of iSCSI with the case in which they are not optimized in NIC. As a result, performance improvement is about 5% with 32ms RTT, and 8% with 64ms RTT.

7. Detailed analysis of data sending in iSCSI storage access

7.1. Overall performance comparison with theoretical value

"sg_dd"[12] command can access target with the specified size of the block at SCSI level in iSCSI access. Since

”sg_dd” command cannot be used in a normal environment, our kernel is reconfigured suitable for the command. By the reconfiguration of our kernel, ”sg_dd” command enables us to access target with the size of the 4096KB maximum. In accordance with the change, in iSCSI parameters, values of both FirstBurstLength and MaxBurstLength are set to 4,194,304. In our experimental system, we have measured throughput when the access block size is 4MB and RTT is 0-50ms. Figure 8 shows the result. A theoretical value is also shown in Figure 8 for the comparison. The theoretical value is calculated from the following equations.

$$Throughput[byte/s] = \frac{blocksize[byte]}{RTT[s] + send[s]} \quad (1)$$

$$send[s] = \frac{blocksize[byte]}{bandwidth[bps]/8} \quad (2)$$

According to Figure 8, performance of the optimized iSCSI is still about half of theoretical performance. To improve the performance of the optimized iSCSI closer to theoretical performance, in this section, the cause of the performance deterioration in long latency has been investigated.

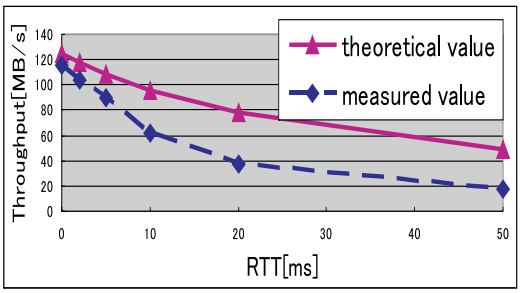

Figure 8. Execution chart of iSCSI write access

7.2. Sequence of ISCSI write access

Figure 9 shows the sequence when iSCSI write access of 4MB is executed. *Ta* means data transfer time of sending from the first packet to the last one. *Tb* is time to prepare a packet to inform the end of writing on Target side to Initiator side. *Tc* is an interval until the following write is executed. *Ta*, *Tb*, *Tc*, and RTT has been measured by the network analyzer with each RTT set at the network simulator. According to a measurement result, *Tb* and *Tc* are almost constant and RTT is almost the same with the value set at the network simulator. On the other hand, the reason why the performance of iSCSI access decreases below the theoretical value is that the data transfer time of sending(*Ta*), which is constant generally, is proportional to RTT. Thus, what happens during data transfer time should be examined in detail.

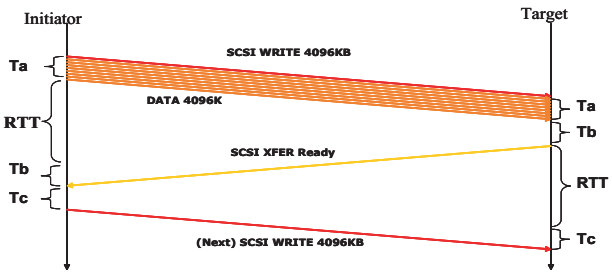

Figure 9. Write process of iSCSI

7.3. Evaluation of CWND

It is known that the values of throughput and CWND have a close relationship. We have used the TCP CWND monitor tool and “ tcpdump ” command, and observed the relationship between CWND and the amount of sent packets. In this experiment, we have set 50ms RTT and 4MB iSCSI access block size. Figure 10 shows the result.

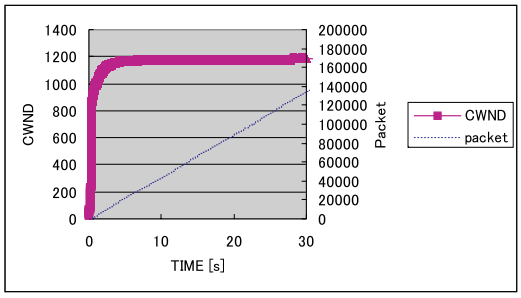

Figure 10. Amount of sent packet and CWND

In order to transfer 4MB data efficiently, about 3000 CWNDs is necessary, since 4096KB/1.448KB= 2828.7. However, according to Figure 10, CWND is about 1200 and the size is not enough for efficient data transmission.

7.4. Detailed analysis of data sending

For more detailed analysis, we have observed packets with the network analyzer. In this experiment, we have set 50ms RTT and 4MB iSCSI access block size. Figure 11 shows the result.

According to Figure 11, the following behavior can be observed. After packets are transferred consecutively within short time, the sending of packet stops suddenly. After constant time, the consecutive sending of packets starts again. Apparently this is the cause of performance deterioration of iSCSI in a long-latency environment.

Those intervals are about 50ms, which is equal to RTT. However, the amount of sending packets is about 600,which is not enough amount that exhausts CWND. In addition, it has been observed that only TCP ACKs are received before the restart of data sending according to more detailed packet analysis with network analyzer.

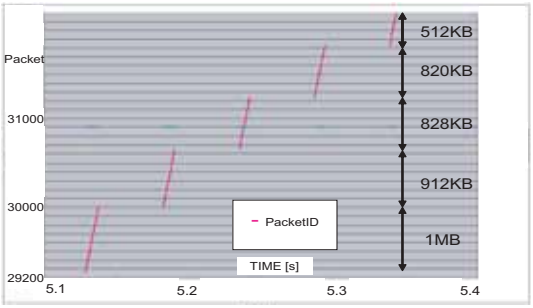

Figure 11. Packet analysis of iSCSI

7.5. Discussion

As a result of the analysis, the following feature has been confirmed. The measured CWND is smaller than the value which can transfer 4MB packets effectively. The amount of sending packets in one sequence is not enough size that consumes CWND. The sending of the packets is intermittent, and this should be the cause of performance deterioration of iSCSI in a long-latency environment. Only TCP ACKs are received before the restart of data sending. Therefore, the conclusion should be as follows. The suspend and restart of iSCSI access should be controlled in the TCP layer because only TCP ACKs are received before the restart. However, the control is not decided only with CWND because this is not consumed completely. Currently, we are investigating the behavior further.

8. Conclusion

In this research, optimization of iSCSI remote storage access through multiple layers has been done, for the purpose of getting the better performance of iSCSI. When RTT is 32ms, performance of the optimized iSCSI has achieved quadruple of that of the default iSCSI. Moreover, We have constructed the model of iSCSI sequential access and analyzed CWND and packet transfer for the purpose of analysis of the cause of the performance deterioration in long-latency. As a result, the following conclusions are obtained. The suspend and restart of iSCSI access are controlled in the TCP layer because only TCP ACKs are received before the restart. However, the control is not decided only with CWND because this is not consumed completely.

As a future work, we will investigate the behavior of inside of iSCSI access in long-latency. We will optimize the system using such a result.

References

[1] http://acs.lbl.gov/TCP-tuning/linux.html

[2] W. T. Ng, B. Hillyer, E. Shriver, E. Gabber and B. Ozden: "Obtaining High Performance for Storage Outsourcing," *Proc. FAST 2002, USENIX Conference on File and Storage Technologies*, pp.145-158, January 2002.

[3] P. Sarkar, S. Uttamchandani, and K. Voruganti: "Storage over IP: When Does Hardware Support help?," *Proc. FAST 2003, USENIX Conference on File and Storage Technologies*, pp.231-244, January 2003.

[4] S. Aiken, D. Grunwald, A. R. Pleszkun and J. Willeke: "A Performance Analysis of the iSCSI Protocol," *Proc. 20th IEEE/NASA Conference on Mass Storage Systems and Technologies (MSST2003)*, pp.123-135, April 2003.

[5] G. Motwani and K. Gopinath: "Evaluation of Advanced TCP Stacks in the iSCSI Environment using Simulation Model," *Proc. 22nd IEEE/NASA Conference on Mass Storage Systems and Technologies (MSST2005)*, pp.210-217 (2005).

[6] A. Joglekar, M. E. Kounavis, and F. L. Berry: "A Scalable and High Performance Software iSCSI Implementation," *Proc. FAST 2005, USENIX Conference on File and Storage Technologies*, pp.267-280, December 2005.

[7] Y. Shastry, S. Klotz, and R. D. Russell: "Evaluating the Effect of iSCSI Protocol Parameters on Performance," *Proc. IASTED International Conference on Parallel and Distributed Computing and Networks (PDCN2005)*, 456-085, February 2005.

[8] B. K. Kancherla, G. M. Narayan, and K. Gopinath: "Performance Evaluation of Multiple TCP connections in iSCSI," *Proc. 24th IEEE Conference on Mass Storage Systems and Technologies (MSST2007)*, September 2007.

[9] K. Hiraki, M. Inaba, J. Tamatsukuri, R. Kurusu, Y. Ikuta, H. Koga, and A. Zinzaki: "Data Reservoir: Utilization of Multi-Gigabit Backbone Network for Data-Intensive Research," *Proc. SC2002*, November 2002.

[10] http://dast.nlanr.net/Projects/Iperf/

[11] http://www.textuality.com/bonnie/intro.html

[12] http://sg.torque.net/sg/sg3_utils.html/