Performance Evaluation of iSCSI System Optimized for Encryption Processing in the Upper Layer

Kikuko Kamisaka†, Saneyasu Yamaguchi‡, Masato Oguchi†

† Graduate School of Humanities and Sciences
Ochanomizu University
2-1-1, Otsuka, Bunkyo-ku, Tokyo, Japan

‡ Institute of Industrial Science
The University of Tokyo
4-6-1, Komaba, Meguro-ku, Tokyo, Japan

Abstruct

iSCSI protocol, used in building IP-based storage networks, is becoming more important because it realizes consolidation of storage at low cost. Security is a critical issue for the iSCSI protocol, on which remote storage is accessed over the IP networks. iSCSI can employ IPsec that offers a function of strong encryption. However, IPsec encryption processing degrades the performance of storage access and increases the CPU load of the server.

In this paper, to perform secure storage access on iSCSI networks, we propose the storage access method of encrypting in the upper layer and its optimization, instead of using IPsec encryption that leads to the performance degradation. We implemented our proposed system, and experimentally evaluated our system accessed with multi process for simulating optimization of encryption pre-processing. As a result, our proposal, optimizing of encryption pre-processing is more efficient than the method using IPsec.

1 Introduction

Through the spread of a broadband network, large-volume multimedia contents are transferred on the network and processed on a server machine in recent years. With a huge volume of data and management cost, Storage Area Network (SAN) is attracting a growing interest. SAN is a high-speed network used to connect servers to storages, thus it allows the storage to be consolidated and managed in a centralized manner. With the advent of broadband LAN technologies such as Gigabit Ethernet, IP-SAN becomes common gradually. IP-SAN technology uses TCP/IP networks, thus it makes administration easy and keeps management costs low, and provides seamless integration with existing IP networks.

Internet SCSI (iSCSI) protocol, ratified by the IETF in February 2003, is expected to become a dominant IP-SAN protocol in the near future. In iSCSI, a SCSI command is encapsulated into TCP/IP packets and transferred between a server(initiator) and a storage(target) via IP networks. iSCSI protocol stack has a complex hierarchical structure, which is SCSI over iSCSI over TCP/IP over Ethernet.

In iSCSI, one of the key issues is a security measure to access storage via IP networks. One of the benefits of using iSCSI is that IPsec is supported. IPsec offers encryption and authentication functions of IP packets. However, IPsec executes the encryption processing in a lower-level, IP layer, and it does nothing but encrypts data segments sequentially, which is passed from an upper layer. Hence IPsec cannot encrypt data effectively. Since there is a trade-off between security and performance, iSCSI communications are required to execute encryption effectively.

In this paper, for realizing secure storage access using iSCSI, we propose an encryption scheme in which transferred data is encrypted in a upper layer instead of the IPsec layer to improve performance, and implement our proposed system. It allows executing pre-processing of encryption, in which a next data segment is encrypted while one data segment is transmitted. Moreover, our proposed system enables to execute flexible processing in the upper layer.

We evaluate the performance of sequential storage access on iSCSI networks in our system. In this experiment, for confirming the effectiveness of encryption pre-processing in the upper layer, we evaluated iSCSI storage access using multi process. In these results, a throughput and CPU load for multi process are improved, as we verify the effectiveness of our proposed method.

2 Issues of Applying IPsec on iSCSI Networks

iSCSI can use IPsec, which offers strong encryption for a secure access to TCP/IP based storage. IPsec offers encryption and authentication functions of IP packets in the IP layer. It commonly employs safe and secure symmetric-key cryptography algorithm, Triple Data Encryption Standard (3DES).

However, storage access using iSCSI requires sending and receiving large volumes of data. Since 3DES encryption processing needs a large amount of calculations, it degrades the performance of communication and burdens the CPU with a heavy load.

We have experimentally measured the throughput and the CPU utilization of iSCSI sequential read access using IPsec[1]. As a result, in the case of using IPsec the throughput degrades compared with the case of unencrypted transmission and the CPU utilization is saturated. We have also analyzed the experimental result, and verified that because IPsec encrypts in the IP layer located on a lower-level, it does not effectively execute encryption processing. Since the IPsec is located on the lower-level layer, it only performs

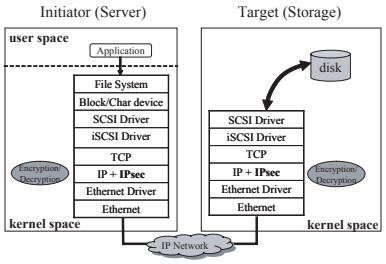

Figure 1. iSCSI Storage Access using IPsec

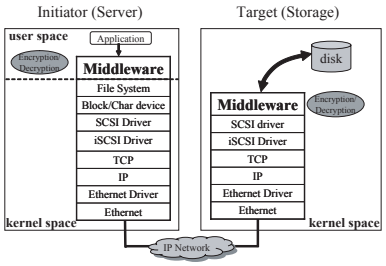

Figure 2. iSCSI Storage Access using Our Proposed System

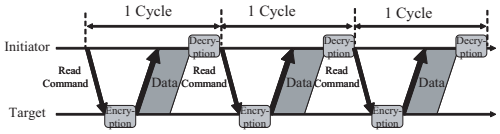

Figure 3. iSCSI Cycle using IPsec

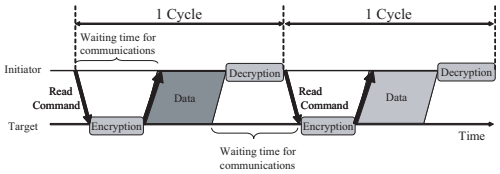

Figure 4. Encryption in the Upper Layer

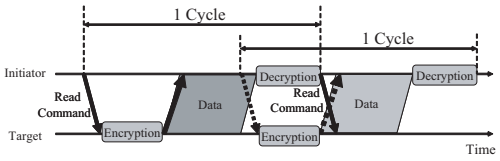

Figure 5. Optimization of Encryption Pre-processing

the data encryption and the IPsec header processing sequentially for small data segments which is passed from the upper layer. In fact, data read from the target's disk is passed to the TCP layer, and those data segments are fragmented into Maximum Segment Size (MSS) in the TCP layer. The IPsec encrypts fragmented data segments and processes the IPsec header for them sequentially. Therefore, this is identified as the cause of overhead and thus it causes the performance degradation.

3 Proposal of iSCSI Storage Access Optimized for Encryption Processing

When IPsec is used on iSCSI communications, it is difficult to encrypt data segments effectively, because IPsec encrypts data segments that is fragmented into a smaller size in the IP layer.

We propose an encryption scheme implemented as middleware in the upper layer than IP layer to access storage on IP-SAN securely, instead of an IPsec encryption scheme[2][3][4]. Figure 1 shows a storage access method in the case of encryption using IPsec on iSCSI networks, and Figure 2 shows a storage access method using our proposed system that uses the encryption scheme in the upper layer.

In the access method using IPsec (Figure 1), data segments stored in the target's disk is passed from the upper layer to the IP layer, and they are sequentially encrypted and decrypted in the IPsec layer after fragmented into a small size. In this way, in the case of using IPsec, since the encryption is executed only serially in the lower layer, it is difficult to execute encryption flexibly such that the next data segment is encrypted while transferring cipher text to the initiator.

In contrast, our proposed scheme (Figure 2) that uses the encryption in the upper layer enables to deal with data segments in a large size efficiently. Moreover, our scheme can greatly reduce a fragmentation overhead of IPsec header addition.

Figure 3 shows iSCSI cycle when IPsec is used. Because IPsec encrypts data segments fragmented into a smaller size, its encryption processing is inefficient.

On the contrary, figure 4 shows iSCSI cycle when data segments are encrypted in the upper layer in our middleware. Since our middleware enables to encrypt data segments by the tally, it greatly reduces fragmentation overhead.

In addition, figure 5 shows iSCSI cycle when a function of optimizing of encryption pre-processing is added to our middleware. In fact, by adding the function to our system, it can encrypts and decrypts the next data segments during waiting time for communications by overwrapping the iSCSI encryption cycle (Figure 5). As encryption time might be hidden in the waiting time of data transmission, the system performance is assumed to improve. Our proposed system that implements encryption optimization in the upper layer enables to deal with processing of application, SCSI, TCP flexibly.

In this paper, to evaluate availability of the encryption optimization shown in figure 5, we issue multiple iSCSI read commands by multi process, so as to simulate the encryption optimization.

Table 1. Experimental system : Spec of Computers

OS	initiator : Linux 2.4.18-3 target : Linux 2.4.18-3
CPU	Intel Xeon 2.4GHz
Main Memory	512MB DDR SDRAM
HDD	36GB SCSI HD
NIC	Intel PRO/1000XT Server Adapter on PCI-X (64bit, 100MHz)

Table 2. Experimental system : iSCSI implementation

iSCSI	UNH-iSCSI Initiator and Target for Linux ver. 1. 5. 3

4 Evaluation of Our Proposed System using multi process

In this paper, we evaluate iSCSI sequential read access to the target's raw device using our system. In this experiment, we implemented the encryption and decryption processing as middleware based on our proposal. This is implemented as a library in an user space in the initiator and as a kernel module in the target. In the target of our system, we implemented an encryption and decryption proprietary program based on 3DES encryption algorithm. Although our implemented algorithm is equal to that of IPsec, there are some differences between our 3DES implementation code and that of IPsec.

To evaluate the effectiveness of the optimization of encryption pre-processing that begins to encrypt the next data before one iSCSI encryption cycle ends (Figure 5). Because we did not implement the pre-processing function on SCSI level, we perform an experiment with multi process so that we can issue multiple iSCSI read command. In the target, the application in the initiator issues up to 5 iSCSI read commands. They access data segments stored in different physical address of target's disk, encrypt in continuity, transfer through a TCP/IP connection, and decrypt in the initiator. In fact, in this experiment, we simulate and evaluate the function of the encryption optimized for pre-processing.

4.1 Experimental Setup

The experimental system consists of the server (initiator) and the storage (target) connected with the Gigabit Ethernet. iSCSI reference implementation offered from the University of New Hampshire InterOperability Laboratory was used[5][6]. Table 1 and 2 show the experimental environment.

4.2 Result and Consideration of Experiments

Figure 6 shows throughput of iSCSI sequential read access when multi process is employed. When 1 process is issued, throughput stays unchanged against difference of a block

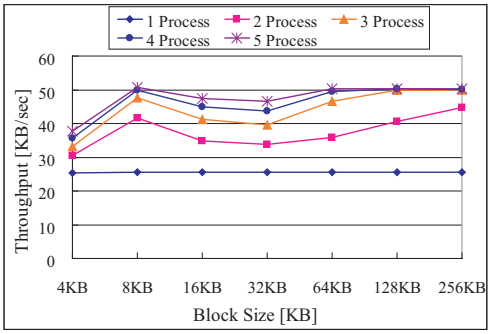

Figure 6. Throughput of Our Experiments with Multi Process

Table 3. Average Throughput Improvement Rate

Process	1	2	3	4	5
Improvement rate	1.000	1.465	1.722	1.812	1.864

size. Table 3 shows average throughput improvement rate with multi process when a block size is changed from 4KB to 256KB. The throughput improvement rate goes up to about 1.5 in 2 processes, about 1.7 in 3 processes. Even if the number of processes is further increased, throughput stays unchanged. This is due to a limit of CPU performance.

In this experiment, since encryption processing time in the kernel module is much longer than communication time, performance improvement is modest at best. However, we assume that optimization of encryption pre-processing based on our proposed method further improve performance of iSCSI storage access in a long-latency environment and in high-performance CPU environment.

Figure 7 shows CPU utilization of iSCSI sequential read access with multi process. The CPU utilization is measured every second by "iostat" at the initiator. It makes no difference between the CPU utilization measured in the initiator and the target. It shows that as the number of process increases, CPU utilization increases as well.

Table 4 shows an average of CPU utilization when the block size is changed from 4KB to 256KB. The CPU utilization is about 51% with 1 process, and it increases up to about 75% with 2 processes. With 5 processes, the CPU utilization achieves to be about 94% and reaches the limit of CPU performance.

5 Throughput Modeling

We analize throughput of iSCSI sequential read access as a measure of performance evaluation of our proposed method.

1 cycle time means that Initiator issues SCSI Read command, encrypts data segments, transferes to the target and they are decrypted in the target. The following equation is relational expression of 1 cycle time

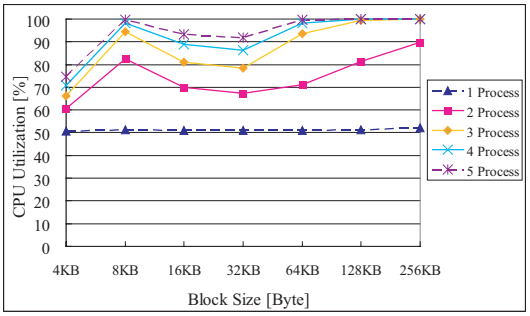

Figure 7. CPU Utilization of Our Experiments with Multi ProcessiInitiatorj

Table 4. Average of CPU Utilization

Process	1	2	3	4	5
Averagei%j	51.241	74.623	87.597	91.760	94.166

(1CYCLE_TIME)C data transfer time (TRANSFER)CRTT (Round Trip Time)Cencryption time (ENC) and decryption time (DEC).

$$1CYCLE\_TIME = RTT + TRANSFER + ENC + DEC \quad (1)$$

The data transfer time is represented as equation (2) using a transfer data size (DATA) and throughput of the lower layer (SOCKET).

$$TRANSFER = \frac{DATA}{SOCKET} \quad (2)$$

When the block size is 256KB, throughput of the lower layer measured experimentally is about 58.106MB/sec, the data transfer time is 4.302msCRTT is 0.392ms and decryption throughput is 50.461KB/sec. Therefore, the throughput is represented as equation (3) using block size (BLOCK).

THROUGHPUT=

$$\frac{BLOCK}{RTT + \frac{BLOCK}{SOCKET} + \frac{BLOCK}{ENC\_TH} + \frac{BLOCK}{DEC\_TH}} \quad (3)$$

Consequently, calculated throughput is 25.219KB/sec with block size of 256KB. Compared to the actual measurement, 25.560KB/sec (Figure 6), it is demonstrated that throughput modeling is almost correct.

On the contrary, As shown in Figure 5, throughput modeling in the optimization in which the next data segment is encrypted during waiting time of communications is represented as equation (4).

$$THROUGHPUT = \frac{BLOCK}{RTT + \frac{BLOCK}{SOCKET} + \frac{BLOCK}{ENC\_TH}} \quad (4)$$

When the block size is 256KB, calculated throughput is 50.414KB/sec. Compared to actual measurement, 44.641KB/sec in 2 processes (Figure 6), the calculated throughput of the optimized model is relatively closeD Consequently, in this experiment, our proposed method that optimizes iSCSI encryption processing cycles by overwrapping demonstrates its effectiveness.

Moreover, if we introduce a high performance CPU, we have potential for further improvement of system's throughput in the case of a larger number of processes. In this experiment, RTT and the data transfer time is almost negligible compared to encryption time. However, in a high-latency environment, waiting time for communication becomes longer, so that the ratio of performance improvement in our proposed method increases, because the encryption optimization is performed effectively.

6 Conclusion

In this paper, for performing a secure storage access on iSCSI networks, we propose the storage access method of encrypting in the upper layer and optimizing encryption, instead of using IPsec encryption that leads to the performance degradation. We implemented our proposed system, and experimentally evaluated our system with multi process. As a result, our proposal, optimization of encryption pre-processing is more efficient than the method using IPsec.

As a part of future work, we will comprehensively evaluate the performance of our system in a high-latency environment.

Acknowledgment

This project is partly supported by the Ministry of Education, Culture, Sports, Science and Technology, under Grant 13224014 of Grant-in-Aid for Scientific Research on Priority Areas.

References

[1] Yamaguchi, S., Oguchi, M. and Kitsuregawa, M.: iSCSI Analysis System and Performance Improvement of Sequential Access in a Long-Latency Environment, *IEICE Transaction on Information and Systems*, Vol. J87-D-I, No. 2, pp. 216–231 (2004).

[2] Kamisaka, K., Yamaguchi, S. and Oguchi, M.: A Proposal of Performance Improvement in Secure Storage Access using IP-SAN, *Information Technology Letters*, Vol. 3, No. LD-003, pp. 59–61 (2004).

[3] Kamisaka, K., Yamaguchi, S. and Oguchi, M.: A Proposal of System Software in Secure Storage Access using iSCSI, *IPSJ SIG Technical Reports, 2004-OS-97, SWoPP2004*, pp. 97–104 (2004).

[4] Kamisaka, K., Yamaguchi, S. and Oguchi, M.: Performance improvement of an iSCSI-based secure storage access, *the 16th IASTED International Conference on Pararellel and Distributed Computing and Systems (PDCS 2004)* (Gonzalez, T.(ed.)), IASTED, pp. 522–527 (2004).

[5] InterOperability Lab
in the University of New Hampshire,
http://www.iol.uhn.edu/consortiums/iscsi/.

[6] iSCSI Draft,
http://www.ietf.org/internet-drafts/draft-ietf-ips-iscsi-20.txt.