PERFORMANCE IMPROVEMENT OF AN ISCSI-BASED SECURE STORAGE ACCESS

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

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

In this paper, for realizing secure storage access efficiently on iSCSI networks, we propose the idea of an encryption scheme in the higher-level layer instead of an IPsec encryption scheme. We measured the performance on simple socket communication and on iSCSI communication using the proposed model, and compared our proposed scheme with IPsec. Consequently, our proposed method of encryption in the higher-level layer outperforms that of encryption using IPsec.

KEY WORDS
SAN, IP storage, iSCSI, IPsec, Encryption, Sequential access

1 Introduction

With the rapidly growing volume of data and management cost in recent years, Storage Area Network (SAN) is attracting growing interest. SAN is high-speed networks used to connect servers to storages, and allows the storage to be consolidated and managed in a centralized manner. Fibre Channel (FC), one of the most popular storage area networks currently, is a dedicated network. However, with the advent of broadband LAN technologies such as Gigabit Ethernet, large amounts of data are stored across IP networks. Using IP-SAN technology makes administration easy and keeps management costs low.

Internet SCSI (iSCSI) protocol, ratified by the IETF in February 2003, is expected to become a dominant IP-SAN protocol in the near future[1][2]. However, because iSCSI networks employ the TCP/IP suite of protocol which is not necessarily designed for high-speed communications, serious performance issues might be raised on the iSCSI-based storage access in a broadband network. In addition, whereas iSCSI can use IPsec, which offers strong encryption in the network layer of the Internet protocol stack for providing secure communications, IPsec encryption increases the processing load on the server's CPU and degrades its performance. This result comes from the encryption of data block fragmented into small size in a lower-level layer.

In this paper, for realizing secure storage access using iSCSI, we propose an encryption scheme in which transferred data is encrypted in a higher-level layer instead of the IPsec layer to improve performance. First, a preliminary experiment was performed on simple socket communication without using iSCSI. We compared the performance of encryption in the IPsec layer and that in higher-level layer. In this experiment, throughput of the latter is superior to the former at an increase of 30%. Moreover, we evaluated the performance of sequential storage access on iSCSI networks. This experiment simulates a part of our proposed scheme. In both experiments, our proposed encryption process executed in the user mode outperforms that of using IPsec in the kernel mode and reduces the CPU load. In these results, our proposed scheme outperforms IPsec, because data block with a larger size can be encrypted in the higher-level layer effectively.

2 Background

2.1 An Overview of iSCSI

Methods for accessing storage have been evolving gradually from the traditional Direct Attached Storage (DAS), in which a server to storage device using I/O bus, to SAN. SAN is a high-speed dedicated networks that connects multiple storage devices to a server. Using SAN technology provides storage consolidation and centralization, which enables management of storage easily and effectively.

Currently, FC-SAN which uses the network techniques of Fibre Channel (FC) is common as a method for the establishment of SAN. However, it has issues including distance limitation, high hardware cost such as FC's switch or host bus adapter, and the limited number of FC engineers. With the recent advent of broadband common-purpose networks such as Gigabit Ethernet, IP-SAN that uses TCP/IP networks to connect servers and storage devices has been proposed. IP-SAN, compared with FC-

SAN, provides seamless integration with existing IP networks and reduces introduction and operational costs.

iSCSI is a newly emerging block-level protocol of IP-SAN and was ratified by the IETF in February 2003. In iSCSI, a SCSI command is encapsulated in TCP/IP packets and transferred between a server (initiator) and a storage device (target) via IP networks. Since standard SCSI commands are embedded in iSCSI, users can operate a remote storage device directly as if they were accessing to a local disk connected to the server directly.

However, TCP/IP protocol is designed for common-purpose communications. In the case of storage access using iSCSI on a broadband network, the issue of high CPU load caused by TCP protocol processing such as memory copy and interruption, is pointed out.

2.2 Issues of Applying IPsec on iSCSI Networks

In iSCSI, one of the key issues is a security measure to access storage via IP networks. One of benefits of using iSCSI is IPsec support, which is a reliable security technique on the Internet. IPsec offers encryption and authentication functions in the network layer of the Internet protocol stack.

Because IPsec can employ safe and secure Triple Data Encryption Standard (3DES) as an encryption scheme, iSCSI enables secure storage access over IP networks. However, since 3DES encryption processing needs a large amount of calculations, it degrades the performance of communication and burdens the CPU with a heavy load.

In the evaluation of sequential read access on iSCSI networks using IPsec, significant performance degradation occurs[3]. We have analyzed the behavior of TCP layer by visualizing TCP packet transfer[4]. According to the result, when SCSI read command has been issued, a set of TCP packets are transferred to the server (initiator) almost simultaneously, then ACK is sent back to the storage (target). However, using IPsec, because the encryption of TCP packets takes long time, they are transferred one by one with regular intervals. This is identified as the cause of the performance degradation.

Since there is a trade-off between security and performance, iSCSI communications require to be contrived to transfer data securely.

In related research works, Wee Tech Ng et al. have studied of iSCSI protocol performance[5]. P. Sarkar et al. have evaluated the performance of iSCSI software and hardware implementation[6][7]. P. Radkov et al. have investigated iSCSI and NFS performance[8]. S-Y. Tang et al. have compared IPsec and SSL schemes on iSCSI protocol[9]. However, security techniques for improving performance using iSCSI are not discussed well until now.

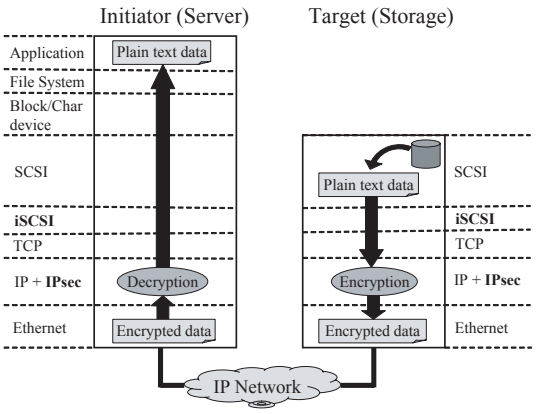

Figure 1. Sequential storage access using the encryption scheme in IPsec layer on iSCSI networks.

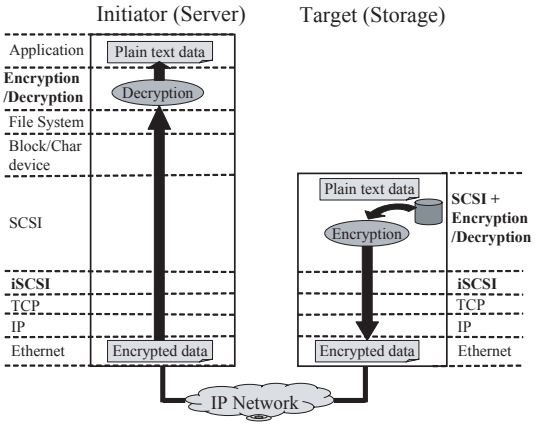

Figure 2. Sequential storage access using the encryption scheme in higher-level layer on iSCSI networks.

3 Proposal of Encryption Scheme in Higher-level Layer for Performance Improvement

When IPsec is used on iSCSI communications, it is difficult to encrypt data effectively, for it cannot comprehend processing in a higher-level layer such as TCP. For instance, in the case of sequential storage access using IP-SAN, IPsec simply encrypt each packet that is passed from TCP layer and it is fragmented into a small size. It is unable to understand processing in the high-level layer so as to encrypt data block fragmented into a large size. It is also impossible to transmit one data block and encrypt next data block while waiting for the ACK.

We propose the idea of an encryption scheme in the higher-level layer to access storage using IP-SAN securely, instead of an IPsec encryption scheme. Our proposal, higher-level encryption access method without IPsec, is illustrated in a figure. Figure 1 shows sequential storage access method in the case of encryption using IPsec on iSCSI networks, and Figure 2 shows sequential storage access

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 and IPsec implementation

iSCSI	UNH-iSCSI Initiator and Target for Linux ver. 1. 5. 3
IPsec	FreeS/WAN ver. 2.01

method in the case of encryption in the higher-level layer.

In the access method using IPsec (Figure 1), data block stored in the target disk is passed from high-level layer to IPsec layer, and it is encrypted and decrypted in IPsec layer after fragmented into small size.

In contrast, in our access method (Figure 2), data stored in the target disk is read, passed to the encryption/decryption layer located in the SCSI layer and encrypted. Encrypted data in SCSI layer is passed to IP and Ethernet layer and transferred to the initiator. The initiator decrypts to a plain text in the encryption/decryption layer.

Performing encryption involves header processing for each fragmented blocks, which is a factor in performance degradation. Encrypting blocks in a large size in higher-level layer is more efficient.

4 Evaluation of Our Proposal Method Using Modeling System

In this paper, as an evaluation of the performance of our proposed method, we have measured throughput of sequential read access for comparing encryption schemes executed in the IPsec layer and in the higher-level layer. In this experiment, we performed simple read access using sockets on networks environment that does not include iSCSI, as a preliminary experiment. Next, we also evaluated sequential storage access on iSCSI networks.

4.1 Experimental Setup

The experimental system consists of the server (initiator) and the storage (target) connected with the Gigabit Ethernet. As an access method of encryption, 3DES cryptographic algorithm was used in both proposed method and IPsec. In the case of encryption in the higher-level, we used OpenSSL crypto library that implements a wide range of cryptographic algorithms in various Internet standards. iSCSI reference implementation offered

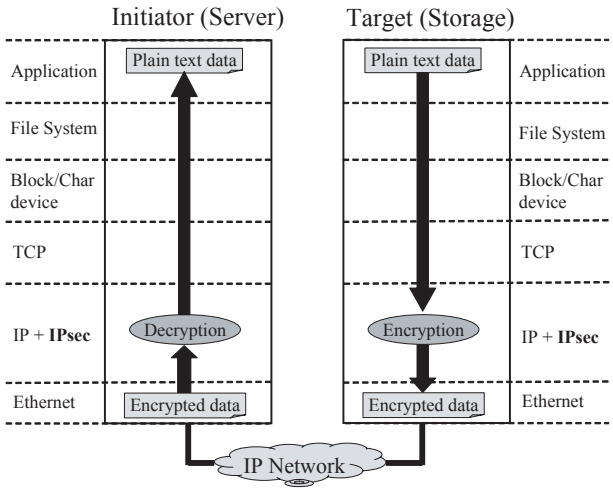

Figure 3. Experiment of simple socket communication using IPsec

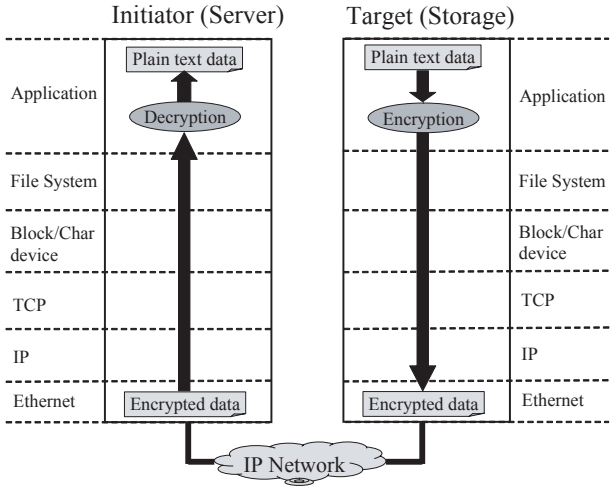

Figure 4. Experiment of simple socket communication with encryption in the higher-level layer

from the University of New Hampshire InterOperability Laboratory[10][11] was used. As an IPsec implementation, FreeS/WAN for Linux [12] was used. IPsec was set up with Transport Mode used to encrypt a host-to-host communication. In this experiment, IP header is not encrypted in both proposed method and IPsec.

In iSCSI networks, a significant performance degradation is caused when a tiny 0.5KB iSCSI Protocol Data Unit (PDU) is issued by Linux implementation of UNH-iSCSI[3]. The tiny packet starts the Nagle algorithm and the delayed-Ack, which is one of the factors of performance degradation. To prevent this phenomenon, we set TCP_NO_DELAY option that disables the Nagle algorithm. Table 1 and 2 show the experimental environment.

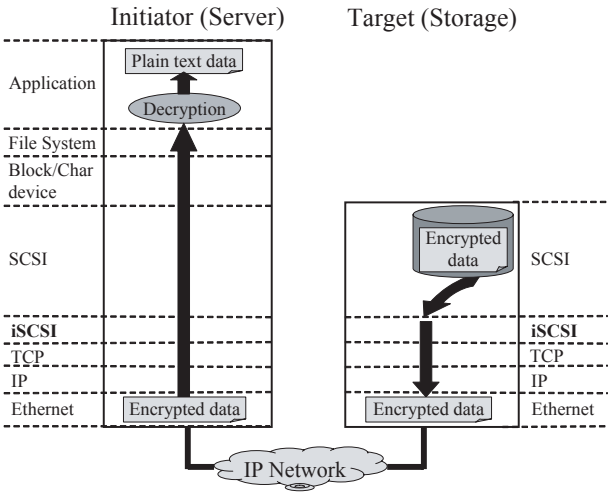

Figure 5. Experiment of iSCSI storage access based on the proposed model

4.2 Experiment of Sequential Read Access in Simple Socket Communication

First, we measured the performance of simple read access using sockets that does not include iSCSI for estimating the magnitude of the performance impact by encrypting data in the higher-level layer. In both encryption schemes executed in the IPsec layer and in the higher-level layer, data blocks are read from a host disk (equals to the target's disk), transmitted using sockets over IP networks, and received by the other host (equals to the initiator).

In the case of access method using IPsec (Figure 3), an application located on the higher-level layer simply executes transmission of data blocks and does not execute encryption processing. Encryption and decryption processing is executed in the IPsec layer located on lower-level. In the case of access method using encryption in the higher-level layer (Figure 4), an application located on the higher-level layer of a host encrypts data stored in the host's disk by calling a cipher procedure in the crypto library. The encrypted data passed through the lower-level layer, normal TCP/IP, is transferred to the other host and decrypted in the application layer.

4.3 Experiment of Sequential Storage Access on iSCSI Networks

Next, we experimentally performed sequential read access to the target's Raw device using iSCSI. In this experiment, our proposed model that executes encryption in the higher-level layer is simulated for estimating the performance of the system implemented actually.

In the case of using IPsec, a normal sequential read access is performed (Figure 1). Data is encrypted at the target and decrypted at the initiator during the communication. In contrast, in our model, encrypted data using the crypto library is stored at the target disk in advance. In sequential read access, the initiator sends a request for reading the encrypted data from the target's disk, and the data is transferred to the initiator and decrypted in the application layer in higher-level (Figure 5). Because the initiator reads data that is previously encrypted and stored at the target in our model, we do not make a fair comparison of two access methods for encryption. In this experiment, however, we use the model for estimating the performance of our proposal's ideal case, in which encryption processing time might be hidden in the waiting time of communications.

5 Result and Consideration of Experiments

5.1 Throughput Comparison

Figure 6 and 8 show throughput results measured in the cases when transferred data is encrypted in the IPsec layer and in the higher-level layer. In the first experiment, the performance of simple socket communication was measured (Figure 6). As a result, the method of encryption in the higher-level layer is superior in terms of throughput, and it achieves 12.3MB/sec on average (an increase of 30%).

This is because that the proposed method encrypting a number of data blocks together in the higher-level layer is more efficient than the method encrypting each blocks of small size in the IPsec layer. In the second experiment, the performance of sequential storage access on iSCSI networks based on our proposed model was evaluated (Figure 8). According to the figure, higher throughput is achieved 10.4MB/sec on average (an increase of 17%). Although cryptographic codes of IPsec and OpenSSL are not completely the same, this comparative experiment is almost fair, because a large amount of calculation is required for the secure 3DES encryption, and the calculation is dominant in the total execution.

5.2 CPU Utilization Comparison

The CPU utilization reported every second by "iostat" at the initiator is shown in Figure 7 and 9. It shows a breakdown (system, user) of the total CPU utilization.

In the case of simple socket communication (Figure 7), the total CPU utilization of the encryption scheme in the higher level layer is slightly reduced for large block size, compared with that of the IPsec encryption. The total CPU utilization of our encryption method is less than 3% that of IPsec encryption on average and less than 15% at 4MB block size. Similarly, in iSCSI sequential storage access using our proposed model, the total CPU utilization of our encryption method is less than 8% that of IPsec encryption on average.

In both experiments, when IPsec is used, the CPU utilization of a "system" makes up the majority of total usage

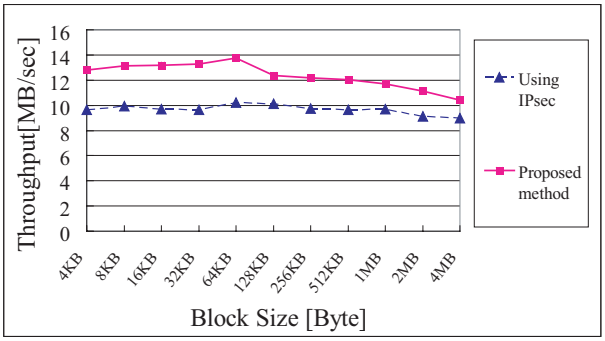

Figure 6. Throughput in the experiment of simple socket communication

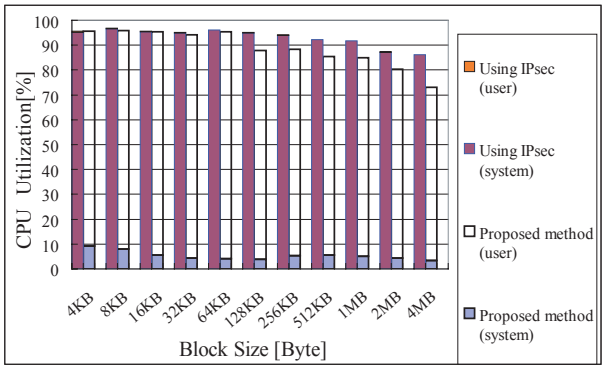

Figure 7. CPU Utilization in the experiment of simple socket communication

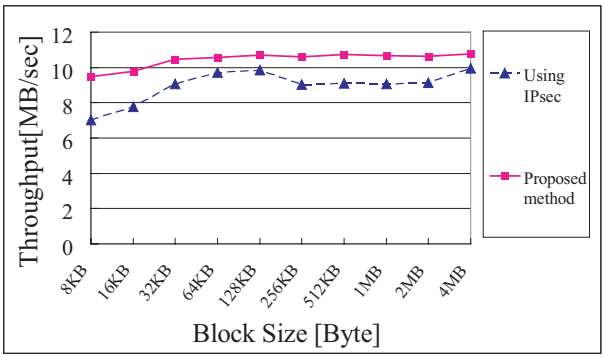

Figure 8. Throughput on iSCSI communications

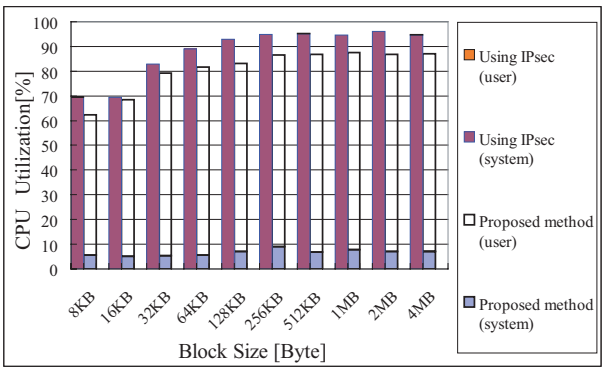

Figure 9. CPU Utilization results in iSCSI communications

and the "user" CPU utilization has little. On the contrary, in the case of communication on iSCSI networks using our proposed model, the "system" CPU utilization is less than 10% and the "user" CPU utilization accounts for a large percentage.

While IPsec encryption is processed in kernel mode, encryption in the higher-level layer is a processed in user mode. In general, the kernel mode process is executed faster because of its higher priority [1]. Nevertheless, in this experiment, the user mode process outperforms the kernel mode process using IPsec, and moreover the CPU load is reduced. The result shows a great improvement in the CPU usage can be achieved by our proposed method. It is desirable to avoid a large amount of processing that leads to high CPU load such as encryption in kernel mode, because other processes in user mode are suspended, which causes the significant performance degradation. The proposed method to encrypt data in the higher-level layer benefits from migrating the process from the kernel mode to the user mode.

As shown in the previous section, we have not yet implemented the encryption scheme in the higher-level layer at the target. However, when this is implemented for secure storage access using iSCSI, the performance of the system can be improved as much as that of ideal case (shown in Figure 8 and 9) by hiding the encryption processing time in the communication time.

[1] We have compared these two modes using simple programs, which is presented in the Appendix

6 Conclusion

In this paper, to perform secure storage access on iSCSI networks, we propose the storage access method of encrypting in the higher-level layer, instead of using IPsec encryption that leads to the performance degradation. We experimentally measured the throughput and the CPU utilization on simple socket communication and on iSCSI communication using the proposed model. Consequently, our proposal, the method of encryption in the higher-level layer outperforms that of encryption using IPsec. We compared the two encryption methods and evaluated them. As a result, our proposed method is more efficient than the method using IPsec.

As part of future work, we will complete the implementation of the encryption processing in the higher-level layer and evaluate its performance.

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] IETF IP Storage (ips) Charter, http://www.ietf.org/html.charters/ips-charter.html

[2] Storage Networking Industry Association, http://www.snia.org/

[3] K. Kamisaka, S. Yamaguchi and M. Oguchi: Analysis of TCP packet transfer on iSCSI storage access using IPsec, *IPSJ SIG Technical Reports, 2004-HPC-97, HOKKE2004*, Hokkaido, Japan, 2004, 145-150.

[4] S. Yamaguchi, M. Oguchi and M. Kitsuregawa, 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*, 2004, 216-231.

[5] 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*, Monterey, CA, 2002, 145-158.

[6] P. Sarkar and K. Voruganti: IP Storage: The Challenge Ahead, *Proc. Tenth NASA Goddard Conference on Mass Storage Systems and Technologies*, 2002, 35-42.

[7] 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*, San Francisco, CA, 2002, 231-244.

[8] P. Radkov, L. Yin, P. Goyal, P. Sarkar and P. Shenoy: Performance Comparison of NFS and iSCSI for IP-Networked Storage, *Proc. FAST 2002, USENIX Conference on File and Storage Technologies*, San Francisco, CA, 2004, 101-114.

[9] S-Y. Tang, Y-P Lu and D. H. C. Du: Performance Study of Software-Based iSCSI Security, *Proc. First International IEEE Security in Storage Workshop*, Greenbelt, Maryland, 2002, 70-79.

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

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

[12] FreeS/WAN Project, http://www.freeswan.org/

Appendix

Comaparison between Kernel Mode and User Mode

Kernel mode processes have higher priority than user mode processes with most OS implementation. Thus kernel mode processes can complete calculation faster than user mode processes. In this section, we present comparison of processing time with kernel mode and user mode of Linux kernel 2.4. Following two experiments were done to compare them: (1)Calculating Fibonacci Numbers using recursive call, (2)Iterating modular operations with two integers. As the results, Figure 10 and 11 are obtained.

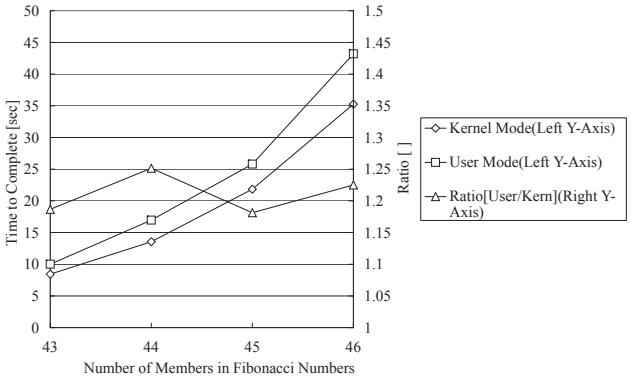

Figure 10. Comaparison of Kernel Mode and User Mode (A) : Fibonacci Numbers

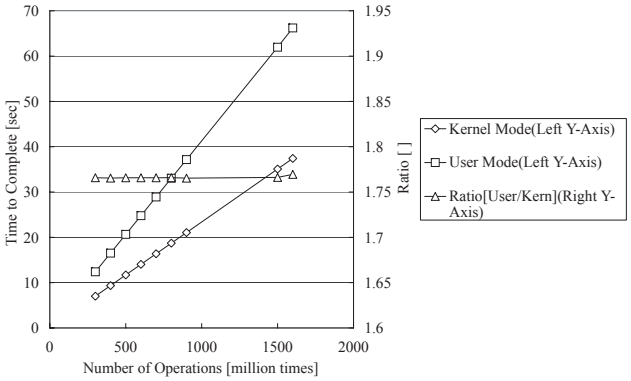

Figure 11. Comaparison of Kernel Mode and User Mode (B) : Modular Operations

X-Axis of Figure 10 stands for the number of members in Fibonacci Numbers, and Y-Axis stands for time to complete calculations. The figure shows that kernel mode processes can complete calculation faster than user mode processes, and the ratios of speedup are about 1.2.

X-Axis of Figure 11 stands for number of repeated modular operations and Y-Axis stands for time to complete them. The figure also shows that kernel mode processes can complete calculations faster than user mode process. The ratios of speedup in this case are about 1.77.

Time to complete is 35.08[sec] with kernel mode and 61.95[sec] in user mode with 1500 million modular operations if they are processed independently. In the case of concurrent execution of the both processes, time to complete in kernel mode and user mode are 38.08[sec] and 61.95[sec] respectively. The result shows that kernel mode processes have higher priority than user mode processes. Kernel mode processes can occupy all CPU resources, while no CPU resources are allocated for user mode process during the execution of kernel mode processes.