Modeling and Improving Security of a Local Disk System for Write-Intensive Workloads

Mais Nijim Xiao Qin*
Department of Computer Science
New Mexico Institute of Mining and Technology
Socorro, New Mexico 87801
{mais, xqin}@cs.nmt.edu

Tao Xie
Department of Computer Science
San Diego State University
San Diego, California 92182
xie@cs.sdsu.edu

Abstract

Since security is of critical importance for modern storage systems, it is imperative to protect stored data from being tampered or disclosed. Although an increasing number of secure storage systems have been developed, there is no way to dynamically choose security services to meet disk requests' flexible security requirements. Furthermore, existing security techniques for disk systems are not suitable to guarantee desired response times of disk requests. We remedy this situation by proposing an adaptive strategy (referred to as AWARDS) that can judiciously select the most appropriate security service for each write request while endeavoring to guarantee the desired response times of all disk requests. To prove the efficiency of the proposed approach, we build an analytical model to measure the probability that a disk request is completed before its desired response time. The model also can be used to derive the expected value of disk requests' security levels. Empirical results based on synthetic workloads as well as real I/O-intensive applications show that AWARDS significantly improves overall performance over an existing scheme by up to 358.9% (with an average of 213.4%).

1. Introduction

In the past decade, storage systems have been an object of substantial interest because of an increasing number of emerging data-intensive applications like video surveillance [2], long running simulations [34], digital libraries [33], remote-sensing database systems [8], and out-of-core applications [24]. This trend can be attributed to the advances in computational power, disk performance, and high-speed networks. There are many cases where data-intensive applications require enriched security to protect data stored in storage systems from talented intruders [37]. Further, a large number of data-intensive applications require guaranteed response times for interactive or high-priority data [12]. Therefore, storage systems are required to provide strong security and guaranteed response times for disk requests. This demanding requirement is

* Corresponding author. *http://www.cs.nmt.edu/~xqin* A version of this paper appeared in the *ACM Transactions on Storage*, vol. 2, no. 4, pp. 400-423, Nov. 2006.

increasingly becoming a critical and challenging issue in the development of the next generation storage systems.

Although conventional storage systems are aimed at improving access times and storage space, many existing storage systems are vulnerable to a wide variety of potential threats. As such, the existing disk systems fail to meet the security requirements of modern data-intensive applications. To protect storage systems against all possible security threats, researchers have developed various ways of ensuring security of data in storage systems.

In this paper, we seek to present a novel adaptive write strategy for local disk systems providing a diversity of security services with various quality of security. The strategy can be seamlessly integrated into disk scheduling mechanisms in disk systems. The proposed strategy is conducive to the achievement of high security for the local disk systems while making the best effort to guarantee desired response times of requests. To prove the efficiency of the proposed approach, we build an analytical model to measure the expected value of security levels and the probability that a disk request is completed before its desired response time.

The rest of the paper is organized as follows. In the next section we summarize related work. Section 3 describes the model of disk requests and the new architecture of storage systems. In section 4, we propose the adaptive write strategy for security-aware storage systems. We build an analytical model in Section 5. Section 6 and 7 present the experimental results based on both synthetic benchmarks (read/write) and real I/O-intensive applications. Finally, Section 8 concludes the paper with future directions.

2. Related Work

There is a large body of work in improving performance of disks, because disk I/O has become a serious performance bottleneck of computer systems. Previous techniques supporting high performance storage systems include disk striping [5][29][30], parallel file systems [9][18][23], load balancing [24][30], caching and buffering [13][15][19].

Disk scheduling algorithms also play an important role in reducing the performance gap between processors and disk I/O [10][17][31][38]. The shortest seek time first (*SSTF*) algorithm is efficient in minimizing seek times, but it is starvation-bound and unfair in nature [11]. The SCAN scheduling algorithm can solve the unfairness problem while optimizing seek times [11]. Reist and Daniel proposed a parameterized generalization of the SSTF and SCAN algorithms [26]. However, the above disk scheduling algorithms are unable to guarantee desired response

times of disk requests.

Many data-intensive applications require that data is stored or retrieved before a desired response time [27]. The SCAN-EDF can be employed to fulfill this requirement [31]. Recently, many disk schedulers were implemented for a mixed-media data set, a mixture of data accessed by multimedia applications and best-effort applications [3][6]. Several disk-scheduling algorithms were proposed to provide quality of service guarantees to different classes of applications [7][25][32]. The salient difference between the proposed approach and the existing disk-scheduling algorithms in the literature is that our strategy is focused on maximizing security of a local disk. Moreover, our strategy is orthogonal to the existing disk scheduling policies in the sense that the novel strategy can be readily integrated into the existing disk schedulers to improve security of local disks.

In recent years, the issue of security in storage systems has been addressed and reported in the literature. Riedel *et al.* developed a common framework of core functions required for any secure storage system [28]. To protect data in untrusted storage systems, researchers designed and implemented cryptographic file systems where data is stored in encrypted form [4][16]. Several key distribution schemes were proposed in SFS [20] and SNAD system [21]. Although a variety of secure storage systems were implemented, there is no adaptive way of choosing security services to meet disk requests' flexible security requirements. Furthermore, the above security techniques are not suitable for disk requests with desired response times. We remedy this situation by proposing an adaptive strategy that can judiciously choose the most appropriate security service for each write request while making the best effort to guarantee the desired response times of all disk requests.

In our previous work, we proposed a family of dynamic security-aware scheduling algorithms for clusters [35][36] and Grids [37]. Unfortunately, these scheduling algorithms limit their applicability to computing resources and, thus, our previous algorithms can not be employed to storage systems.

3. Architecture and Disk Requests with Security Requirements

3.1 Architecture of a Security-Aware Storage System

In this study we focus on local disk systems, and storage systems in parallel and distributed environments are out the scope of this paper. Our new algorithm is based on a security-aware storage system architecture. Fig. 1. depicts the main components, i.e., a disk driver, a security

mechanism, and a disk scheduling core, of this architecture.

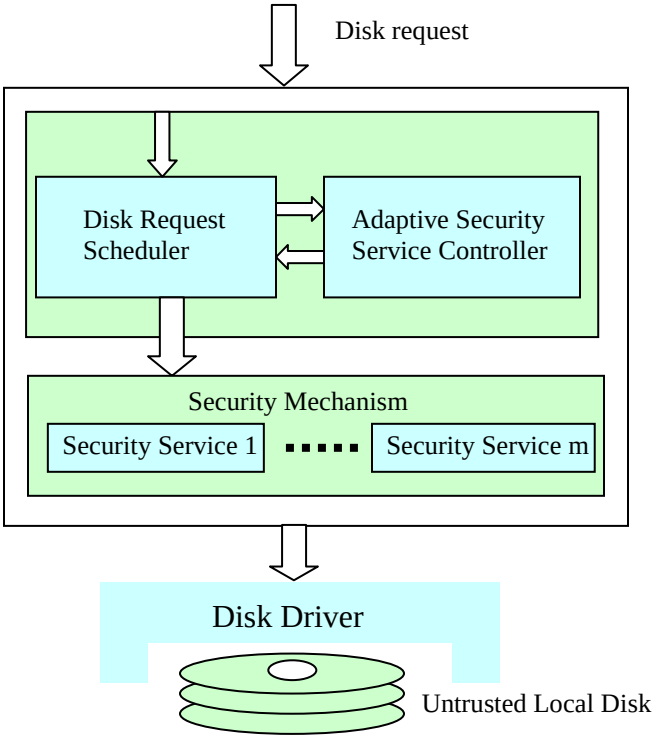

Figure 1. Architecture of a security-aware storage system

The architecture is briefly overviewed as follows. The disk driver is responsible for controlling access to an untrusted local disk. The security mechanism provides an array of security services to guard data blocks residing in the local disk against unauthorized access and information theft. Without loss of generality, we consider in this study only confidentiality security services, where it is assumed that keys are known only to the owner, reader and writers [4]. The security mechanism can be readily extended to employ integrity and availability services. The disk scheduling core consists of two parts: a disk scheduler and an adaptive security service controller. While the scheduler implements generic logic and timing mechanisms for scheduling and waiting, the security service controllor dynamically choose the most appropriate security service for each disk request. Since the security service controller is independent of disk scheduling policies, the service controller is implemented separately for each disk scheduler. As such, it is easy to apply the security service controller to any disk scheduling policy implementation.

3.2 Modeling Disk Requests with Security Requirements

Each disk request submitted a security-aware storage system specifies quality of service requirements, including security and performance requirements. A security requirement is defined as a lower bound security level, which is a value from 0.1 to 1.0. A performance requirement is posed as a desired response time. The quality of service requirements of disk requests can be derived from applications issuing these disk operations. Security quality of encryption services implemented in the security mechanism is measured by security levels. An encryption service with a high security level means the high quality of security provided by the service. For example, a disk request specifies a lower bound security level as 0.4. In this case, encryption services with security levels higher than or equal to 0.4 can successfully meet the disk request's security requirements.

A disk request r is characterized by five parameters, $r = (o, a, d, s, t)$, where o indicates that the request is a read or write, a is the disk address, d is the data size measured in KB, s is the lower security level bound, and t is the desired response time.

The security benefit gained by a disk request r_i can be measured by the security level σ_i of an encryption service facilitating confidentiality for the disk request. Likewise, the quality of security offered by a local storage system can be measured by a sum of security benefit of all incoming disk requests. Let R be a set of incoming disk requests. Our proposed AWARDS strategy strives to maximize the security benefit of the storage system. Thus, we can obtain the following non-linear optimization problem formulation to maximize the security benefit, where ρ_i is the real response time of the ith disk request.

$$\text{Maximize } \sum_{r_i \in R} \sigma_i$$
$$\text{Subject to } \forall r_i \in R : s_i \le \sigma_i \le 1, \text{and } \rho_i \le t_i . \quad (3.1)$$

3.3 Security Overhead Model

Now we consider security overhead incurred by confidentiality services. The security overhead model can be easily extended to incorporate other security services. Encryption is used to encrypt data blocks residing in local storage systems. There are ten encryption algorithms (see Table 1) implemented in the security mechanism. Based on the encryption algorithms' performance, each algorithm is assigned a security level. For example, level 0.9 implies that we use 3DES, which is the strongest yet slowest encryption function among the alternatives. Note

that overhead of encryption depends on the chosen cryptographic algorithm and the size of data block. Fig. 2. plots enciphering time in seconds as a function of the encryption algorithms and data size on a 175 MHz Dec Alpha600 machine [22][36][37]. Let σ_i be the security level of r_i, and the security overhead can be calculated using Eq. 3.2, where d_i is the data size and $P(\sigma_i)$ is a function used to map a security level σ_i to the corresponding performance of encryption service listed in Table 1.

$$T_{security}(\sigma_i, d_i) = \frac{d_i}{P(\sigma_i)}. \tag{3.2}$$

Table 1. Cryptographic Algorithms Used for Encryption Service

Cryptographic Algorithms	Security Level, σ	Performance KB/ms, $P(\sigma)$
SEAL	0.1	168.75
RC4	0.2	96.43
Blowfish	0.3	37.5
Knufu/Khafre	0.4	33.75
RC5	0.5	29.35
Rijndael	0.6	21.09
DES	0.7	15
IDEA	0.8	13.5
3DES	0.9	6.25

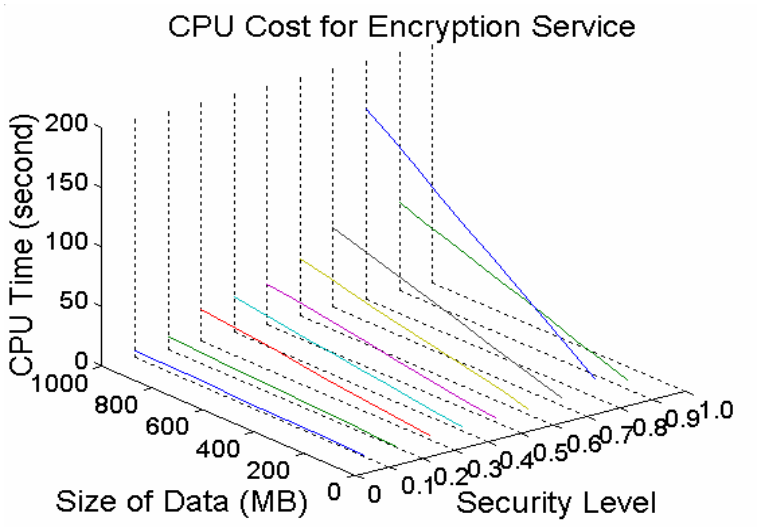

Figure 2. Security overhead of encryption services.

4. The Adaptive Write Strategy

This section presents the proposed adaptive write strategy, which is referred to as AWARDS

throughput this paper. We assume that the overhead of AWARDS is negligible when compared to the processing times of disk requests. In this study we consider a local disk system providing nine encryption services with different security levels (see Section 3.3). AWARDS aims at improving quality of security for local disk systems. To achieve this goal, AWARDS aggressively increases the security level of each incoming disk request under the condition that the request's response time does not exceed the desired response time. We make use of an example to elaborate the basic idea behind the AWARDS strategy. Suppose there are three write requests submitted to a local disk system at time 0. Table 2 shows the important parameters of the three write requests. We assume that the disk bandwidth is 30MB/Sec., and for each request, the sum of rotational latency and seek time is 8ms.

Table 2. Important parameters of the three write requests

Requests	Data Size (d_i)	Minimal Security Level (s_i)	Desired Response Time (t_i)	Response Time (T) under AWARDS	Security Level (σ_i) under AWARDS
r_1	90 KB	0.2	18 ms	17.7 ms	0.8
r_2	150 KB	0.1	41 ms	40.7 ms	0.7
r_3	30 KB	0.3	55 ms	54.5 ms	0.9

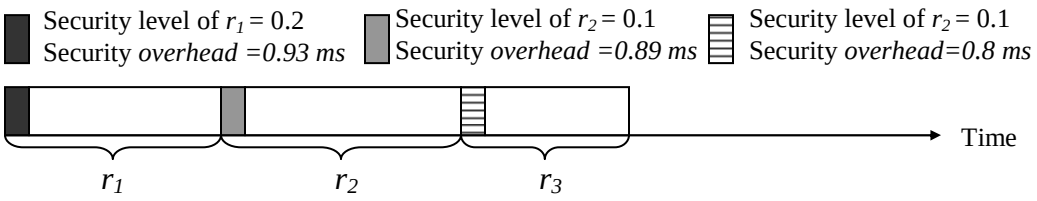

Fig. 3a. Security levels and overhead of the requests. The system is running without AWARDS.

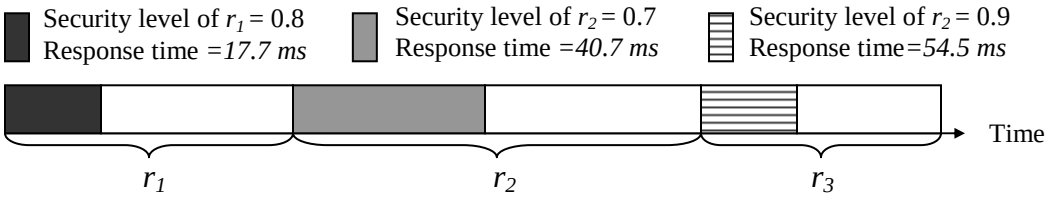

Fig. 3b. Security levels and response times of the requests. The system is running with AWARDS.

Prior to writing the data of a request to the local disk, the disk system enciphers the data using a selected encryption service. Similarly, the system deciphers the data of a read request after the data is retrieves from the disk. Hence, the processing time of each disk request logically consists of two parts: security overhead indicated by a shaded block and disk service time represented by an unshaded block (see Eq. 4.1 and Fig. 3). Figs. 3a illustrates the security levels and response

times of the requests when the disk system is running without AWARDS. In this case, the minimal security requirement of each request is met at the minimal security overhead. In contrast, AWARDS can significantly increase the security levels provided that the desired response times can be guaranteed (see Fig. 3b). For example, the response times of the three requests are 17.7ms, 40.7ms, and 54.5ms, which are less than the respective desired response times (see Table 2). Specifically, with the AWARDS strategy in place, the security levels are improved by an average of 366.7%.

Now we present the AWARDS strategy, which adaptively adjusts security levels of write requests (see Fig. 4). It is worth noting that AWARDS is unable to adjust security levels of read requests, because the local disk system has to use a corresponding encryption service to decipher the data of a read request after the cipher is read from the disk. Before increasing the security level of a write request r_i, AWARDS ensures that r_i and those write requests with earlier desired response times can be finished before their desired response times (see condition 4.2 in the following property). Therefore, the following property needs to be satisfied in the AWARDS strategy.

Property 1. If the security level of a write request r_i is increased by 0.1, the following conditions must hold:

(1) The current security level of r_i is less than 0.9, i.e., $\sigma_i < 0.9$; (4.1)

(2) $\forall r_k \in Q, t_k \geq t_i : \mathrm{es}(r_k) + T(r_k, \sigma_k) \leq t_k$. (4.2)

where Q is the waiting queue, $\mathrm{es}(r_k)$ is the start time of request r_k, and $T(r_k, \sigma_k)$ is the processing time of $r_i \in Q$. The start time $\mathrm{es}(r_k)$ can be expressed by

$$\mathrm{es}(r_k) = \sum_{r_l \in Q, t_l \leq t_l} T(r_l, \sigma_l), \tag{4.3}$$

where the expression on the right side of Eq. 4.3 is the total processing time of disk requests whose desired response time is earlier than that of r_k.

The processing time $T(r_k, \sigma_k)$ in condition 4.2 can be computed by Eq. 4.4.

$$T(r_i, \sigma_i) = T_{seek}(a_i) + T_{rot}(a_i) + \frac{d_i}{B_{disk}} + T_{security}(d_i, \sigma_i), \tag{4.4}$$

where T_{seek} and T_{rot} are the seek time and rotational latency, $\frac{d_i}{B_{disk}}$ is the data transfer time that largely depends on the data size d_i and the disk bandwidth B_{disk}, and $T_{security}(\sigma_i, d_i)$ is the security

overhead that lies in the security level and security-critical data size (see Eq. 3.2).

The adaptive write strategy for secure local disk systems, or AWARDS, is described in Fig. 4. AWARDS aims at improving quality of security and guaranteeing disk requests' desired response times. To achieve high security, the AWARDS strategy optimizes the security levels of write requests (see Step 7).

1. **Input:** r_i, a disk request r_i submitted to the local disk system; Q, a waiting queue
2. Insert r_i into Q based on the earliest desired response time first policy;
3. **for** each write request r_j in the waiting queue Q **do**
4. $\sigma_j \leftarrow s_j$; /* Initialize the security levels */
5. **for** each write request r_j in the waiting queue Q **do**
6. **if** $\sigma_j < 0.9$ **then /*** σ_j can be further increased */ (see **property 1(1)**)
7. Increase security level σ_j by 0.1;
8. **for** each request r_k with longer desired response time **do** (see **property 1(2)**)
9. **if** r_k can not be completed before its desired response time **then**
10. Decrease security level σ_j by 0.1; break;
11. **end for**
12. **if** σ_j is not increased in Step 7 **then** break;
13. **end for**

Figure 4. The adaptive write strategy for local disk systems.

Upon the arrival of a disk request, AWARDS insert the request into the waiting queue based on the earliest desired response time first policy, meaning that disk requests with earlier response time are processed first. Before proceeding to the optimization of security levels of write requests in the queue, AWARDS first initialize the security levels of all the write requests to the minimal levels (see Step 4). Step 7 then gradually enhance the security level of each request r_j under the conditions that (1) the current security level of r_j does not exceed 0.9 (see Step 6); and (2) the desired response times of the requests being processed later than r_j can be achieved (see Steps 8-10). The process of optimizing security levels repeatedly performs (see Step 5); and stops when a request's desired response time can not be met (see Step 12). In doing so, the AWARDS strategy can maximize the security levels of write requests (see Step 7) while guaranteeing the desired response times of all the disk requests in the queue (see Steps 9 and 10). The time complexity of AWARDS is evaluated as follows.

Theorem 1. The time complexity of AWARDS is $O(n^2)$, where n is the number of disk requests in the waiting queue.

Proof. To increase the security level of a request, it takes $O(n)$ time check condition 4.2 (see Step

8). Since there are *O(n)* number of write requests in the waiting queue, the time complexity of optimizing security levels of write requests is: O(*n*)*O(n)* = *O*(n^2).

5. Analytical Model

In this section, we first derive the probability that a disk request is completed before its desired response time. Second, we calculate the expected value of security levels assigned to disk requests with security requirements.

5.1 Satisfied Ratio

We first calculate, for every disk request r_i submitted to a local disk system, the probability that r_i can be finished within the desired response time t_i, i.e., $\Pr(\rho_i \le t_i)$ where ρ_i is the real response time. At any time when a disk request arrives, the request is inserted in the queue in a way that all requests with earlier desired response times will be given higher priority and executed first. Note that *n* waiting requests in the queue are indexed by their priorities so that the desired response time of r_i is smaller than that of r_j if $i < j$, i.e., $\forall 1 \le i, j \le n : i < j \Rightarrow t_i < t_j$. In this study we assume that processing times of different disk requests are statistically independent.

Recall that $T(r_i, \sigma_i)$ is the processing time of a disk request $r_i \in Q$. $T(r_i, \sigma_i)$ is computed by Eq. 4.4. Let p_x be the probability that the disk request r_i requires *x* time unit to complete, i.e., $p_x = \Pr(T(r_i, \sigma_i) = x)$. Similarly, let q_y be the probability that the total required processing time of disk requests with higher priorities is *y*, i.e., $q_y = \Pr\left(\sum_{j=1}^{i-1} T(r_j, \sigma_j) = y\right)$. The probability that r_i is unable to be finished within the desired response time t_i is computed as follows:

$$\begin{aligned}\Pr(\rho_i > t_i) = &\Pr\left(T(r_i,\sigma_i)=1 \,\middle|\, \sum_{j=1}^{i-1} T(r_j,\sigma_j) \ge t_i\right)\\ &\vdots\\ &+\Pr\left(T(r_i,\sigma_i)=k \,\middle|\, \sum_{j=1}^{i-1} T(r_j,\sigma_j) \ge t_i+1-k\right)\\ &\vdots\\ &+\Pr\left(T(r_i,\sigma_i)=t_i+1 \,\middle|\, \sum_{j=1}^{i-1} T(r_j,\sigma_j) \ge 0\right)\end{aligned}$$

$$= p_1 \sum_{y=t_i}^{\infty} q_y + p_2 \sum_{y=t_i-1}^{\infty} q_y + \ldots + p_{t_i} \sum_{y=1}^{\infty} q_y + p_{t_i+1} \sum_{y=0}^{\infty} q_y$$
$$= \sum_{x=1}^{t_i+1} \left(p_x \sum_{y=t_i+1-x}^{\infty} q_y \right), \quad (5.1)$$

where the second line on the above equation indicates the conditional probability that the required processing time of disk request r_i is k given that it requires at least t_i+1-k time unit to complete disk requests with higher priorities.

The probability that a disk request r_i is completed within its desired response time is given by

$$\Pr(\rho_i \le t_i) = 1 - \Pr(\rho_i > t_i)$$
$$= 1 - \sum_{x=1}^{t_i+1} \left(p_x \sum_{y=t_i+1-x}^{\infty} q_y \right) \quad (5.2)$$

5.2 Quality of Security

To evaluate quality of security for a local disk system, we derive in this section the expected security level experienced by disk requests. Before proceeding to the calculation of the expected security level, we compute the probability $\Pr(\sigma_i = z)$ that the security level of each submitted disk request r_i equals to z. Recall that the security level of a disk request relies on the desired response time, data size of the request, and processing times of other waiting requests with higher priorities. As such, $\Pr(\sigma_i = z)$ can be calculated as

$$\Pr(\sigma_i = z) = \Pr(d_i = d_{\min}) \cdot \sum_{j=t_{\min}}^{t_{\max}} \left\{ \Pr(t_i = j) \cdot \Pr\left[\sum_{k=1}^{i-1} T(r_k, \sigma_k) = j - \overline{T}(d_{\min}, z) \right] \right\}$$
$$\vdots$$
$$+ \Pr(d_i = l) \cdot \sum_{j=t_{\min}}^{t_{\max}} \left\{ \Pr(t_i = j) \cdot \Pr\left[\sum_{k=1}^{i-1} T(r_k, \sigma_k) = j - \overline{T}(l, z) \right] \right\}$$
$$\vdots$$
$$+ \Pr(d_i = d_{\max}) \cdot \sum_{j=t_{\min}}^{t_{\max}} \left\{ \Pr(t_i = j) \cdot \Pr\left[\sum_{k=1}^{i-1} T(r_k, \sigma_k) = j - \overline{T}(d_{\max}, z) \right] \right\}$$
$$= \sum_{l=d_{d\min}}^{d_{\max}} \left\{ \Pr(d_i = l) \cdot \sum_{j=t_{\min}}^{t_{\max}} \left\{ \Pr(t_i = j) \cdot \Pr\left[\sum_{k=1}^{i-1} T(r_k, \sigma_k) = j - \overline{T}(l, z) \right] \right\} \right\} \quad (5.3)$$

where $\overline{T}(l, z)$ is the processing time of r_i if the data size is l and security level is set to z, i.e.,

$$\overline{T}(l, z) = T_{seek}(a_i) + T_{latency}(a_i) + \frac{d_i}{B_{disk}} + T_{security}(l, z).$$

The data size and desired response time of each disk request are two random variables distributed according to two probability density functions, which are know *a priori*. We let u_l denote the probability that the data size of the disk request is l, and let v_j denote the probability that the desired response time equals to j. The minimal and maximal data sizes are represented by d_{min} and d_{max}. Likewise, the minimal and maximal desired response times are denoted by t_{min} and t_{max}. Based on Eq. 4.4, the probability $\Pr(\sigma_i = z)$ can be expressed as

$$\Pr(\sigma_i = z) = \sum_{l=d_{d\min}}^{d_{\max}} \left\{ u_l \cdot \sum_{j=t_{\min}}^{t_{\max}} \left[v_j \cdot q_{j-\bar{T}(l,z)} \right] \right\}, \tag{5.4}$$

where $q_{j-\bar{T}(l,z)} = \Pr\left(\sum_{k=1}^{i-1} T(r_k, \sigma_k) = j - \bar{T}(l,z) \right)$.

The expected security level experienced by disk requests with security requirements can be directly derived from Eq. 4.5. Thus, the expected security level is given by

$$E(\sigma) = \sum_{i=1}^{9} \left(\frac{i}{10} \cdot \Pr(\sigma = \frac{i}{10}) \right). \tag{5.5}$$

6. Synthetic Benchmarks

To quantitatively evaluate the performance of the AWARDS strategy, we implemented a basic prototype of AWARDS to work with a simulated local disk system. Workloads with both synthetic benchmarks and real world I/O intensive applications (see Section 7) were generated and evaluated in the prototype of AWARDS.

Our experimental testbed consist of the prototype, the simulated disk system, and the nine encryption services described in Table 2. Disk parameters summarized in Table 3 are similar to those of the IBM Utrastar 36Z15.

Table 3. Disk Parameters.

IBM Ultrastar 36Z15	
Size	18.4 GB
RPM	15000
Seek Time, T_{seek}	7.18 ms
Rotational Time, T_{rot}	4.02 ms
Disk Bandwidth, B_{disk}	30 MB/Sec.

The following four important performance metrics are used to evaluate the AWARDS strategy. (1) *Satisfied ratio* is defined as a fraction of total arrived disk requests that are found to be completed before their desired response times. (2) *Average security level* is the average value

of security levels achieved all disk requests issued to the disk system. (3) *Average security overhead* is measured in seconds. (4) *Overall performance* is measured by a product of the satisfied ratio and average security level.

To provide fair comparisons, we obtained empirical results based on a wide range of synthetically generated workload and environmental conditions, which closely resemble a variety of I/O intensive applications. Table 4 outlines the important workload configuration parameters for the simulated disk system used in our experiments.

Table 4. Workload Configuration

Parameter	Value (Fixed) - (Varied)
Disk Bandwidth	30MB/Sec.
Request Arrival Rate	(0.1, 0.2, 0.3, 0.4, 0.5) No./Sec.
Desired Response Time	10 Sec.
Security Level	(0.5) - (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)
Write Ratio	(100%) - (0%, 10%, 20%, 30%, … 100%)
Data Size	(500 KB) – (300, 400, 500, 600, 700) KB

In Section 6.1 we present performance comparisons between a disk system with AWARDS and another system without employing AWARDS (referred to as the Original strategy). Section 6.2 studies the impact of data size on disk performance. Section 6.3 examines performance sensitivities to disk bandwidth. Performance impact of security requirements is evaluated in Section 6.4. Finally, Section 6.5 demonstrates the performance impact of increasing write ratio.

6.1 Overall Performance Comparisons

This experiment is aimed at comparing AWARDS against Original, which is a strategy without making use of AWARDS. To stringently evaluate the performance of AWARDS and its competitive strategy, we set write ratio to 100%. The impact of write ratio on system performance is studied in Section 6.5. We increased disk request arrival rate from 0.1 to 0.5 No./Sec. Other workload parameters were fixed to the same values as those listed in Table 4.

Figure 5 plots the four performances metrics for the AWARDS and Original strategies. Fig. 1a reveals that AWARDS maintains a very close performance in satisfied ratio to the Original strategy. Fig. 5b shows that AWARDS significantly outperforms Original in average security level by an average of 138.2%. As the request arrival rate increases, the average security levels of the two strategies decrease. However, AWARDS always achieves higher average security levels compared with those of the Original strategy. This result can be explained in part by Fig. 5c, which shows that the average security overhead of AWARDS is constantly higher than that

of the alternative. In other words, high security levels of AWARDS are achieved at the cost of high security overheads. Fig. 5d clearly reveals that AWARDS noticeably outperforms the Original one in terms of overall performance. Specifically, AWARDS obtains an improvement in overall performance over the Original strategy by an average of 125.6%. The performance improvement can be attributed to the fact that AWARDS adaptively enhance security levels of each write requests under the condition that all requests in the disk can achieve their desired response times.

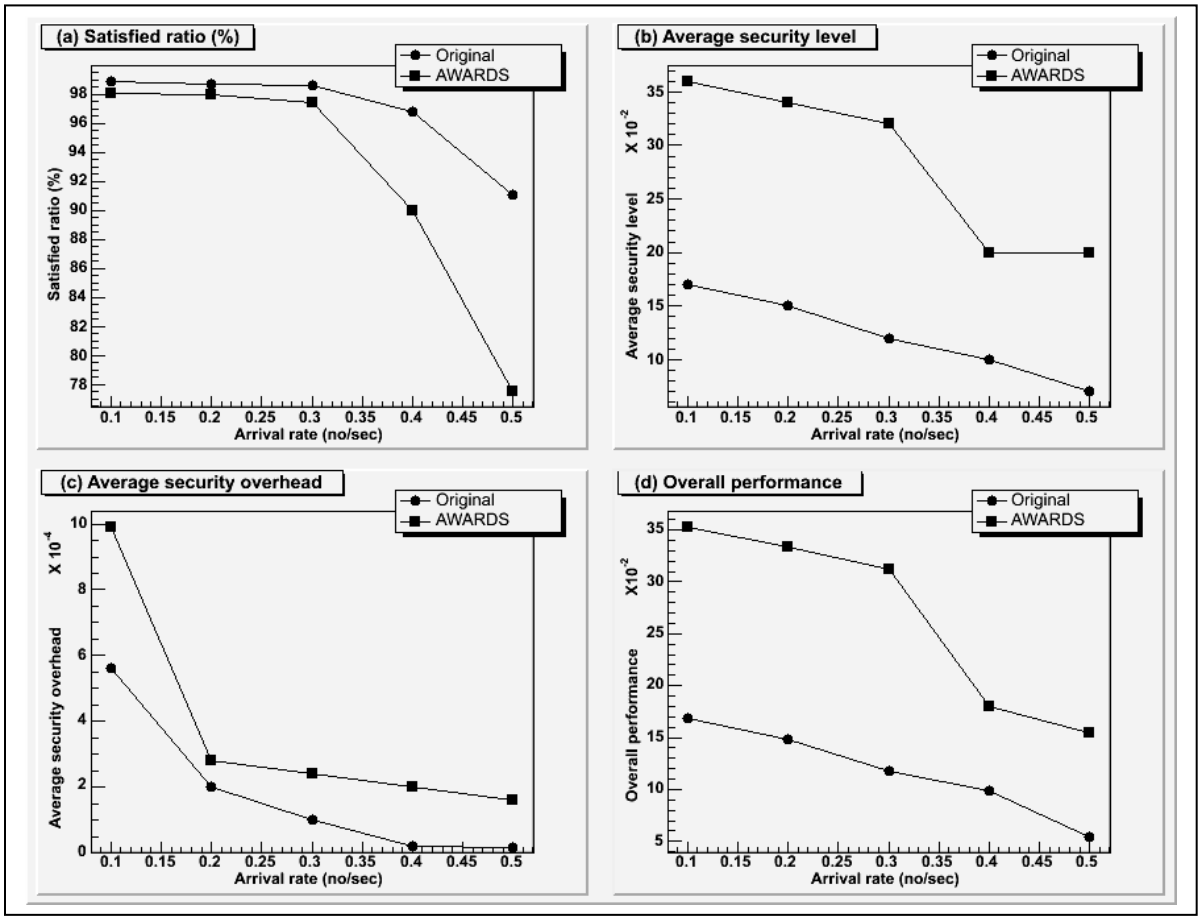

Fig. 5. Performance impact of arrival rate

6.2 Impact of Data Size

In this section, we varied data size from 300 to 700 KB to examine the performance impact of data size on the local disk system. Again, other workload parameters were kept unchanged (see Table 4).

Fig. 6a shows that when the data size increases from 300 to 700 KB the AWARDS strategy delivers similar satisfied ratios as those of Original. This result, which is consistent with the

result presented in Fig. 5a, demonstrates that AWARDS achieves a good performance in satisfied ratio. Like Fig. 5b, Fig. 6b shows a significant improvement of AWARDS in security level over Original.

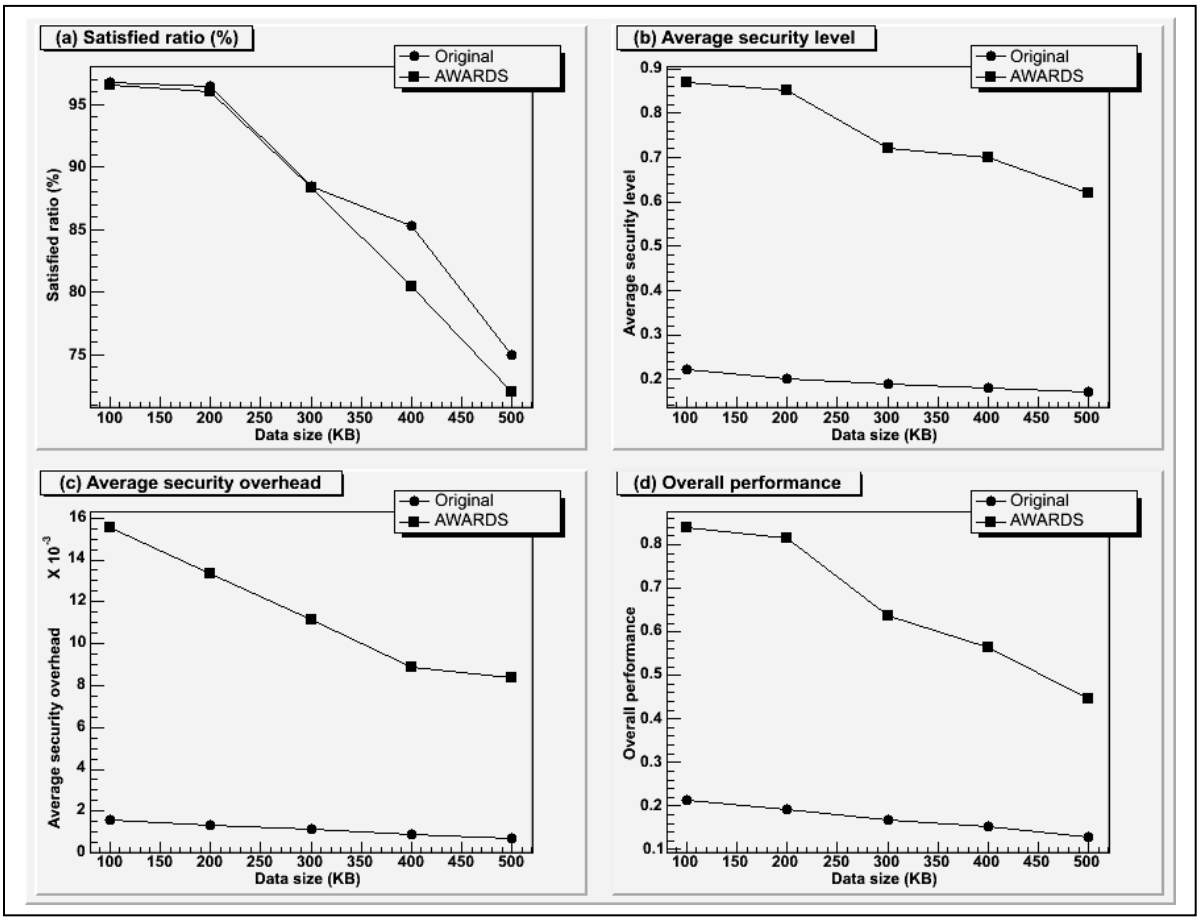

Fig. 6. Performance impact of data size

Interestingly, it is observed from Fig. 6b that the average security level gradually drops with the increasing value of data size. This is because the increasing data size results in a overloaded condition, which in turn leads to decreasing security levels of disk requests in the queue in order to finish most of the requests before their desired response time. Further, we observe from Figs. 6c and 6d that when the data size goes up, the average security overhead and overall performance of AWARDS decreases, because the security levels of disk requests are lowered down due to the high workload. By contrast, the average security level and overhead of the Original strategy only slight reduce with the increasing value of data size. These results indicates that Original is insensitive to data size.

6.3 Impact of Disk Bandwidth

This experiment we investigated the performance of AWARDS and Original when the disk bandwidth varies from 10 MB/sec to 50 MB/sec. An important observation drawn from Fig. 7a is that as the bandwidth increases, the satisfied ratios of the two strategies rise gently. The result can be explained by the fact that the high disk bandwidth leads to short transfer times, which in turn result in short processing times of disk requests. Consequently, more disk requests can be finished before respective desired response times.

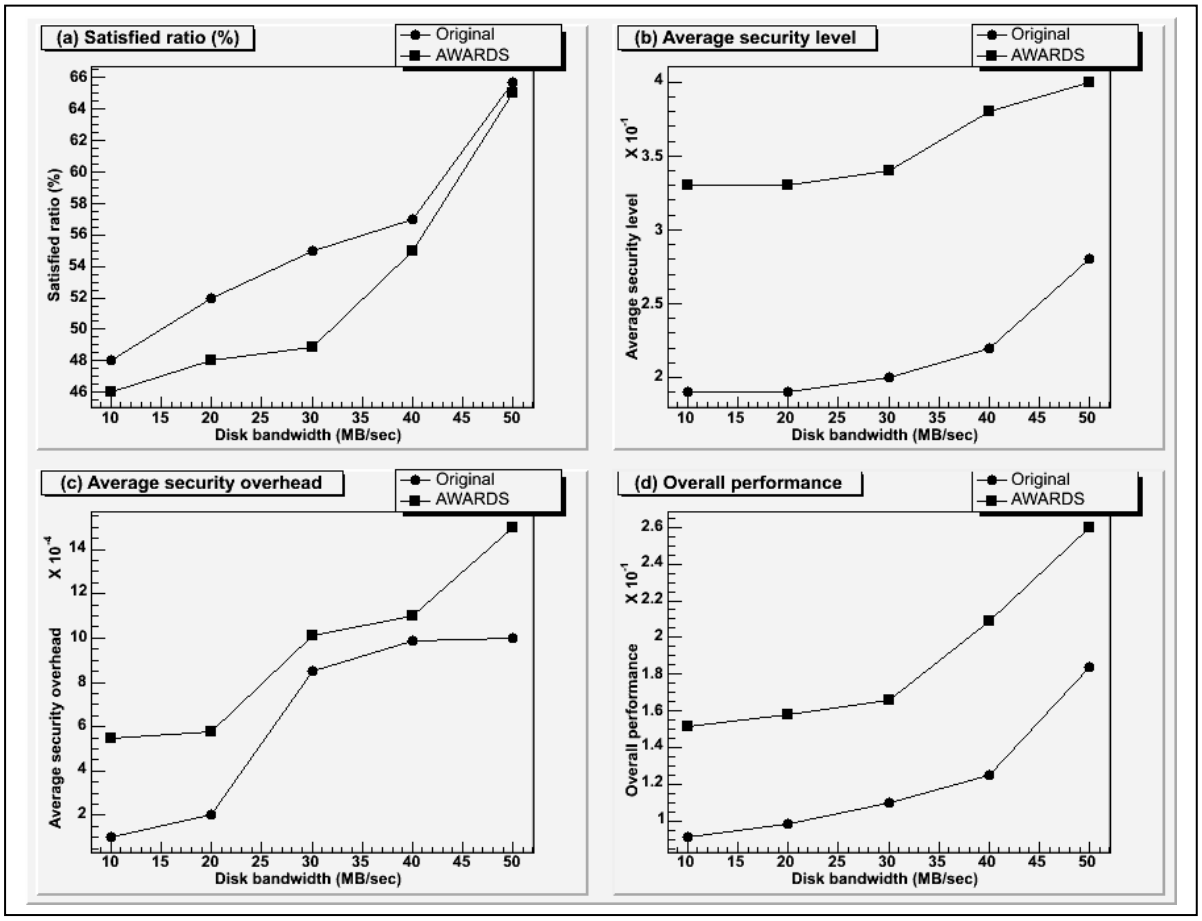

Fig. 7. Performance impact of disk bandwidth

It is worth noting that the satisfied ratio curves of the two alternatives begin to merge when the bandwidth is larger than 40 MB/sec. Fig. 7b shows that the average security level increases as the disk bandwidth is increased, because processing times of disk requests become smaller in the light of high disk bandwidth. The shortened processing times enable AWARDS to further increase security levels at the expense of higher security overheads (see Fig. 7c). Thanks to the increasing satisfied ratio and average security level, the overall performance of AWARDS is substantially boosted in case of a disk system providing high disk bandwidth (see Fig. 7d).

6.4 Security Level Range

In this group of experiments, we studied performance impact of the security requirements of disk requests. The maximal required security level is varied from 0.5 to 0.9, whereas the minimal required security level is fixed at a value of 0.1.

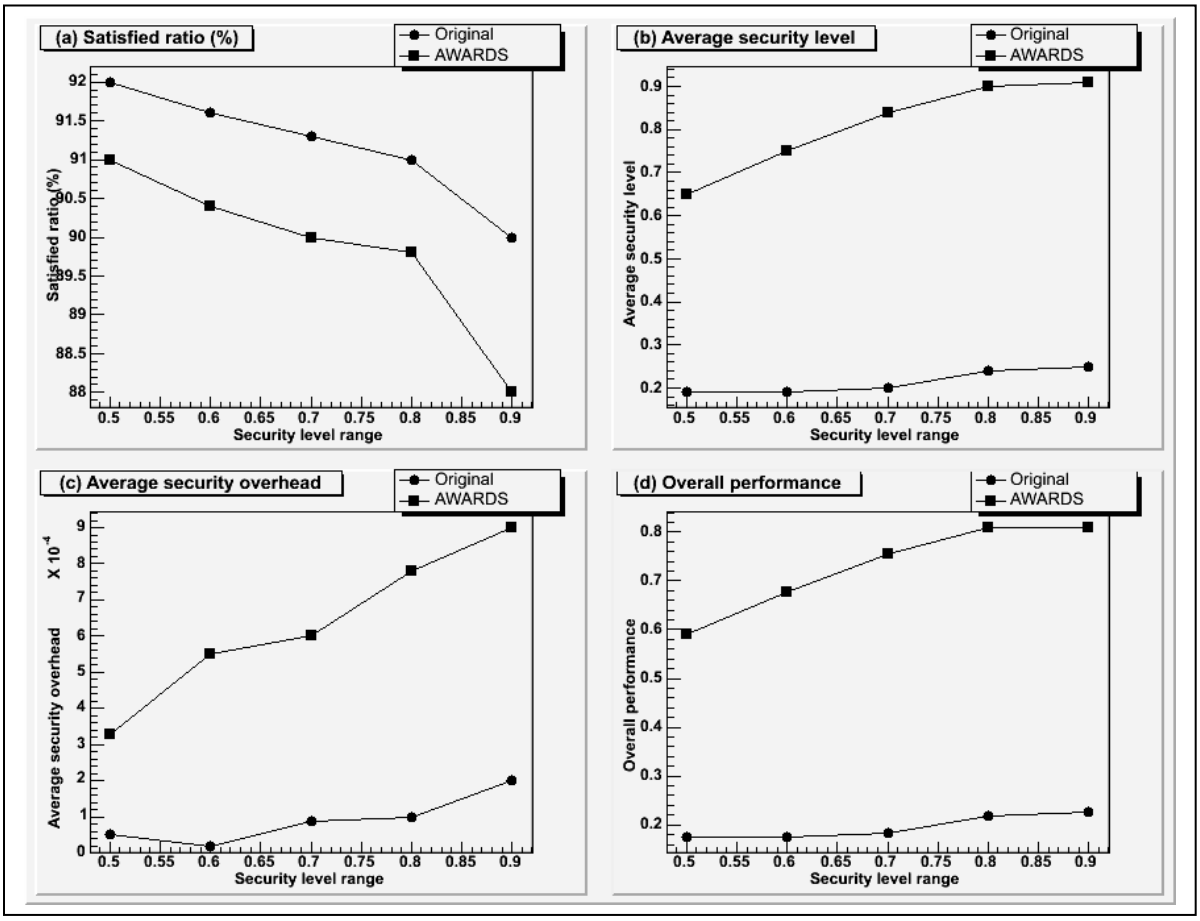

Fig. 8. Performance impact of security level range

It is observed from Fig. 8a that when the maximal required security levels increase, the satisfied ratios of the two strategies decrease. The main reason for this result is that when disk requests require higher security levels, the security overhead is inevitably grows (see Fig. 8c). The growing security overhead in turn causes a significant drop in the satisfied ratio. We observe from Fig 8b that the amount of improvement in average security level becomes more prominent with the increasing value of maximal required security level. This performance trend can be explained by the fact that the larger the maximal required security level, the more opportunities for AWARDS to dynamically increase security level of each write request. Because the average

security level rapidly increases, the overall performance of AWARDS also goes up as the maximal required security levels is increased (see Fig. 8d).

6.5 Impact of Write Ratio

In the previous experiments, we assume that the write ratio is fixed to 100%. This experiment, however, seeks to measure the impact of write ratio on disk performance. We increased the write ratio of the workload from 0% to 100% with increments of 10%. Fig. 9 plots the four performance metrics as functions write ratio.

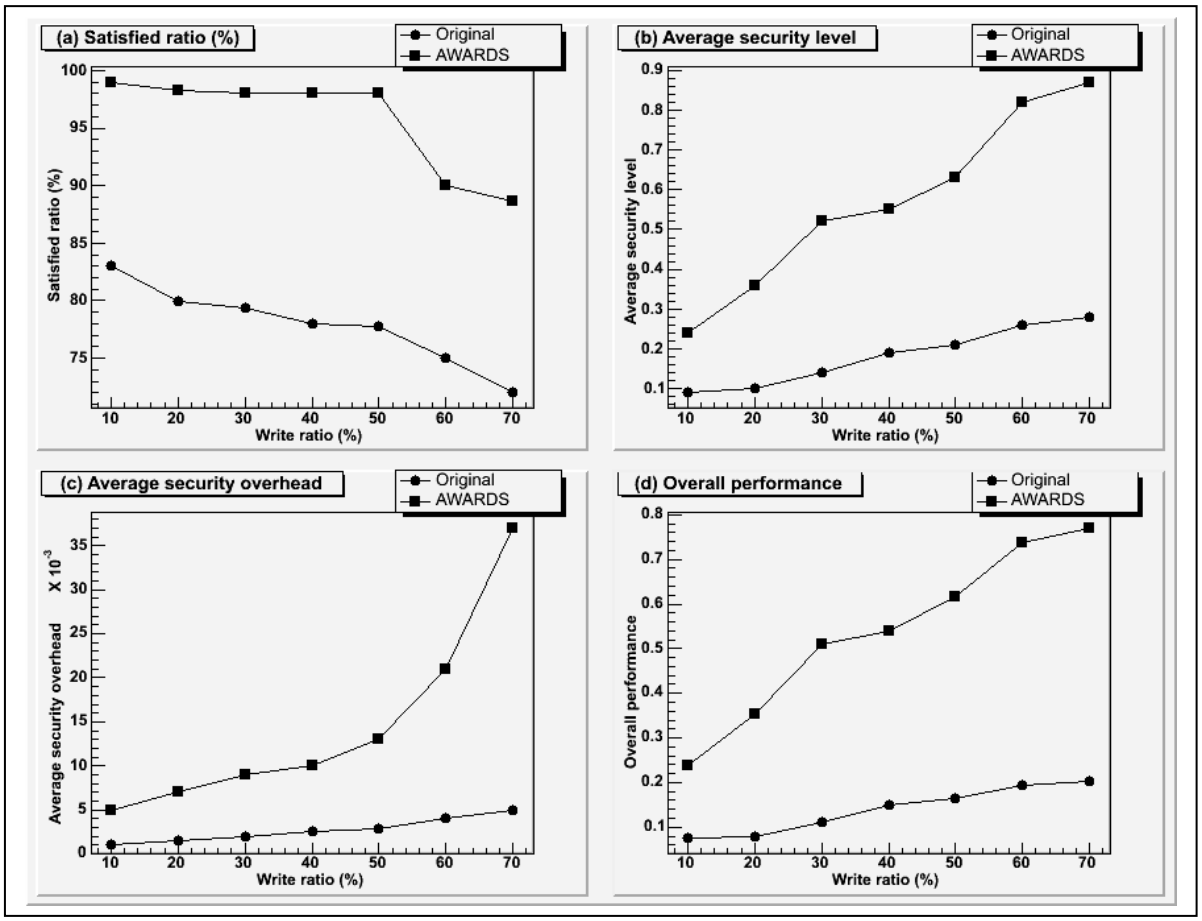

Fig. 9. Performance impact of write ratio

Like the empirical results presented in Figs. 5-8, results shown in Fig. 9 illustrates the performance improvements of AWARDS over the alternative. Fig. 9a shows that the satisfied ratio is marginally reduced with the increasing write ratio, because AWARDS aggressively enhances the security levels of write requests, which experience longer processing times due to higher security overheads (see Fig. 9c). It is intriguing to observe from Fig. 9b that the

performance improvement of AWARDS in security over the Original strategy become more pronounced for higher write ratio. The rationale behind this result is that AWARDS is conducive to the improvement in security for write requests and, thus, increasing number of write requests offers more opportunities for AWARDS to significantly improve security performance by choosing higher security levels for the write requests. Consequently, the overall performance improvement over the competitive strategy is more striking for higher write ratio (see Fig. 9d). This result suggests that the proposed AWARDS approach is suitable for improving security of write-intensive applications like transaction processing, logfile updates, and data collection.

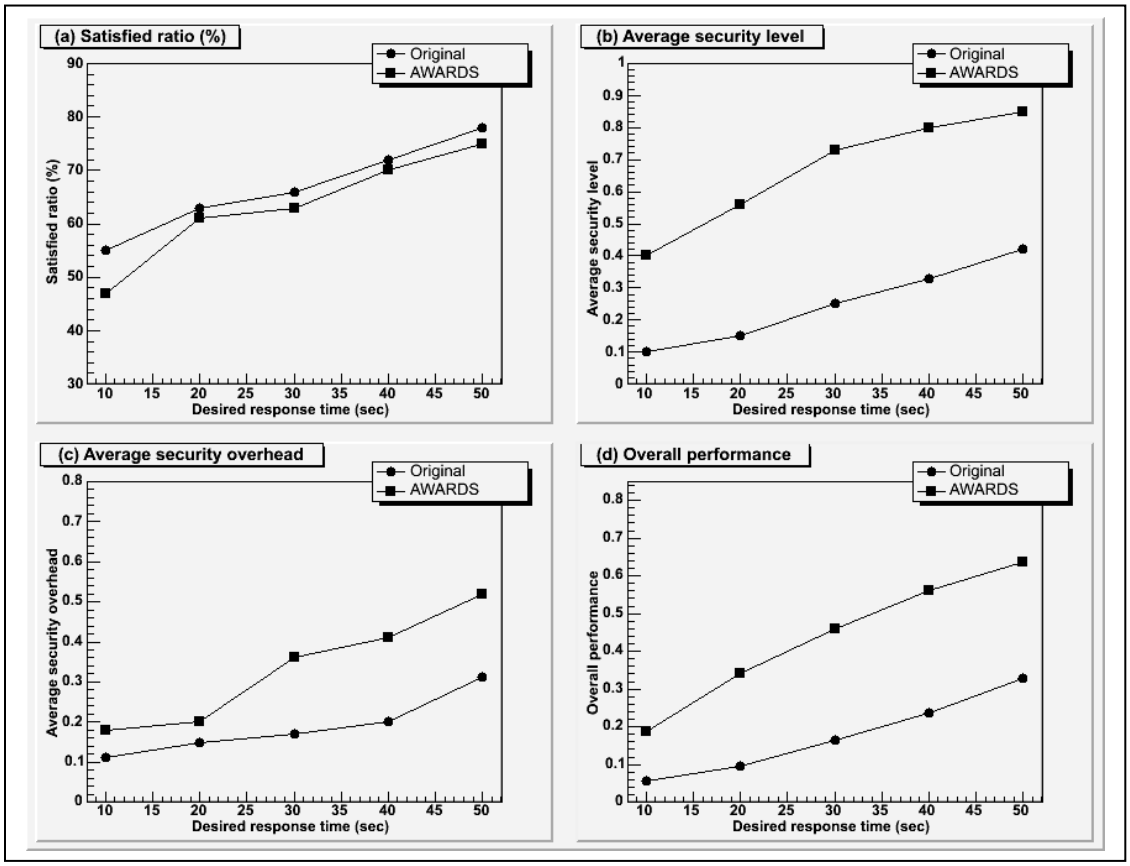

Fig. 10 Performance impact of desired response time. LU Decomposition

7. Real I/O-Intensive Applications

To validate the results from the synthetic workload, we used disk traces of real world I/O-intensive applications to evaluate the performance of our strategy in comparison to the Original scheme. We chose two common I/O-intensive applications: LU decomposition [14] and sparse cholesky [1], which have different I/O patterns. The LU decomposition application tries to

compute the dense LU decomposition of an out-of-core matrix, whereas the sparse cholesky application is used to calculate Cholesky decomposition for sparse, symmetric positive-definite matrices.

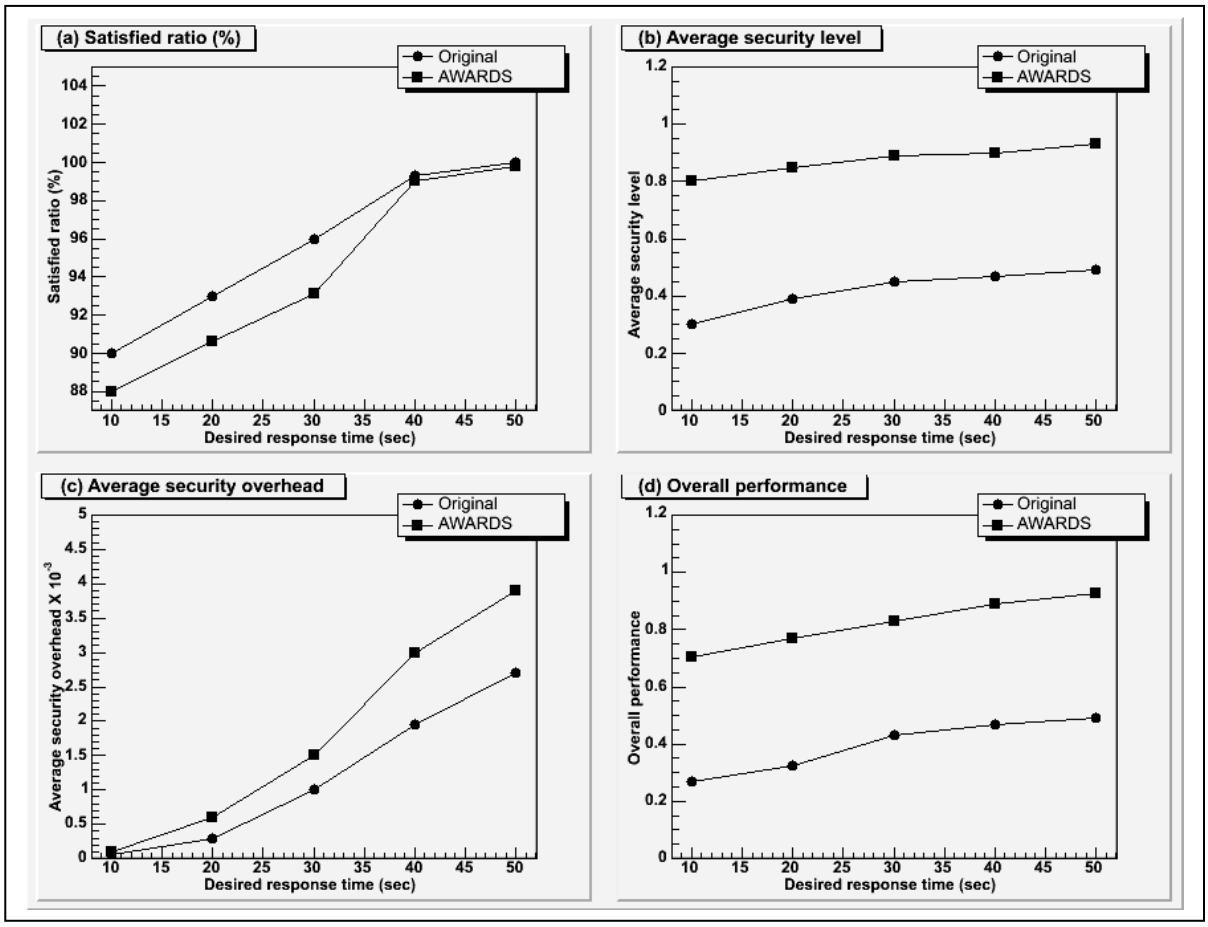

Fig. 11 Performance impact of desired response time. Sparse Cholesky

First of all, we studied how desire response times affect satisfied ratios and security levels. Throughout this set of experiments, the disk bandwidth was set to 30MB/Sec. The results for the LU decomposition and sparse cholesky applications are plotted in Figs. 10 and 11, respectively. Figs 10a and 11a show that for the two I/O-intensive applications, satisfied ratios yielded by AWARDS are close to those of the original strategy. This observation is especially true when the desired response time is long. Figs 10b and 11b demonstrate that as the desired response time increases, the average security levels for all cases increase. This is mainly because when the desired response time is enlarged, the possibility of improving security levels without violating timing constraints increases. This argument is supported by the results summarized in Figs 10c

and 11c, which reveal that the average security overheads for all cases grow with the increase in desired response times. Figs 10d and 11d summarize the overall performance of the two schemes. In general, the performance effect of desired response depends in part on the applications. By comparing the two applications, we observe from Figs 10d and 11d that LU decomposition is more sensitive to desired response time while sparse cholesky is less sensitive. The cause of this performance difference can be explained as follows. The disk request arrival rate and data size are two dominant factors for satisfied ratio and average security level. High arrival rate and large data size of disk requests give rise to LU decomposition's small satisfied ratios and average security levels (See Figs 10a and 10b). Consequently, the increase in desired response time provides more opportunities for LU decomposition to improve both satisfied ratio and average security level, which in turn induce a more pronounced improvement in overall performance when the desired response time is enlarged.

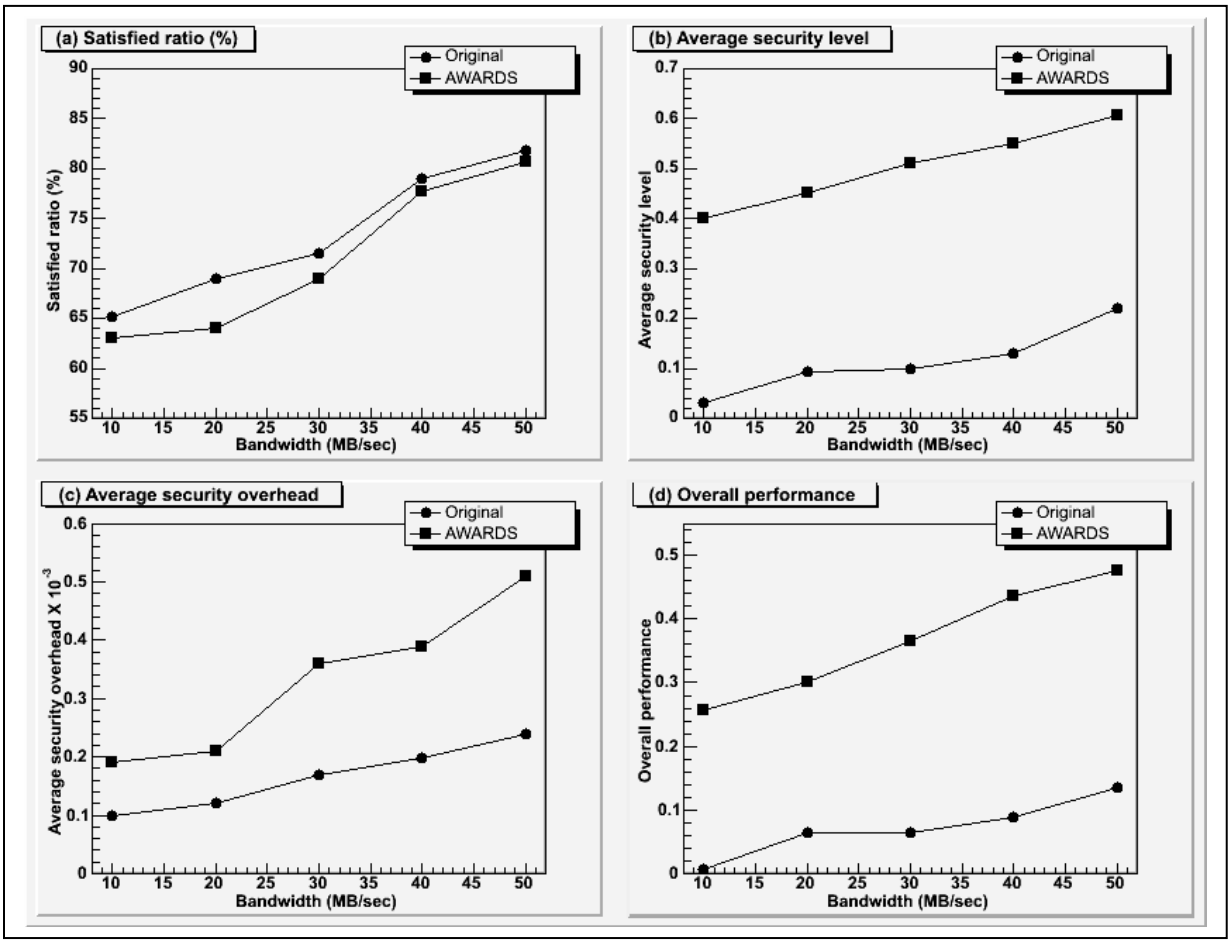

Fig. 12 Performance impact of disk bandwidth. LU Decomposition

Now we evaluate the impact of disk bandwidth on the two real applications. In this group of experiments, the disk bandwidth varies from 10 to 50 MB/Sec with increment of 10MB/Sec. Figs. 12 and 13 depict the disk bandwidth effect on the AWARDS and Original strategies when LU decomposition and sparse cholesky are used as the workload. Figs 12a-12c and 13a-c illustrate that for all the cases we have examined, the increased disk bandwidth is a driving force of the improved satisfied ratios and average security levels. These results are consistent with the disk bandwidth impact seen for the synthetic benchmarks in Section 6.3.

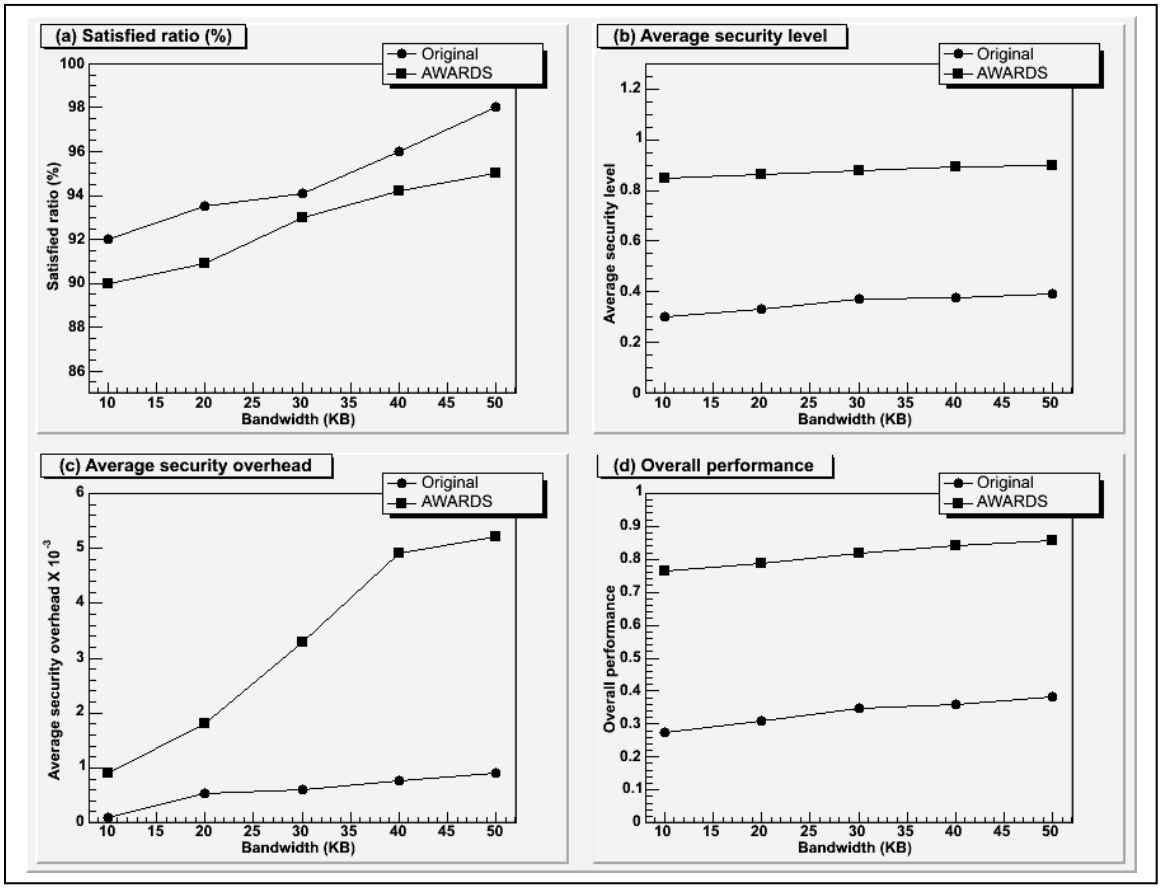

Fig. 13 Performance impact of disk bandwidth. Sparse Cholesky

Not surprisingly, the disk bandwidth effect on the two strategies relies in part on the applications. More specifically, Figs 12d and 13d conclusively show that the overall performance improvement achieved by AWARDS is much more pronounced for the workload with the LU decomposition application as compared to the workload with sparse cholesky. This can be partially attributed to the high sensitivity of LU decomposition to disk bandwidth. LU decomposition compared with sparse cholesky is more sensitive to disk bandwidth, because the

relatively large arrival rate and data size of the workload with LU decomposition induce small satisfied ratios and average security levels, which allow more room for improvement in overall performance.

8. Summary

In this paper, we considered the flexible security requirements of disk write requests in the context of a local disk system. To protect stored data from being tampered or disclosed, we proposed a security-aware storage system architecture. Next, we developed an adaptive write strategy for secure disk systems (AWARDS for short). The AWARDS strategy can adaptively choose an appropriate security service for each write request in a way to maximize the security of the local disk system while making an effort to achieve desired response times of all incoming disk requests. Further, we constructed an analytical model to estimate the probability that a disk request is completed before its desired response time. The model also can be use to derive the expected value of disk requests' security levels. We implemented a prototype of AWARDS and evaluated its performance using synthetic workloads as well as two real world I/O-intensive applications. Experimental results demonstratively show that our strategy outperforms an existing scheme in security and overall performance by up to 325.0% and 358.9% (with averages of 199.5% and 213.4%).

Currently, we are developing and evaluating an extended version of AWARDS for parallel disk systems. Both data placement and load balancing algorithms are being incorporated into the extended version.

Acknowledgments

The work reported in this paper was supported in part by the New Mexico Institute of Mining and Technology under Grant 103295 and by Intel Corporation under Grant 2005-04-070.

References

[1] A. Acharya et al., "Tuning the performance of I/O-intensive parallel applications," *Proc. of the 4th IOPADS*, pages 15–27, Philadelphia, PA, May 1996.

[2] D. Avitzour, "Novel scene calibration procedure for video surveillance systems," *IEEE Trans. Aerospace and Electronic Systems*, Vol. 40, No. 3, pp. 1105-1110, July 2004.

[3] E. Balafoutis, G. Nerjes, P. Muth, M. Paterakis, G. Weikum, P. Triantafillou, "Clustered Scheduling Algorithms for Mixed-Media Disk Workloads in a Multimedia Server," *J. of Cluster Computing*, Vol. 6, No. 1, pp. 75-86, 2003.

[4] M. Blaze, "A Cryptographic File System for UNIX," *Proc. ACM Conf. Communications and Computing Security*, 1993.

[5] R. Bordawekar, R. Thakur, and A. Choudhary, "Efficient Compilation of Out-of-core Data Parallel Programs," *Technical Report, SCCS-662*, NPAC, April 1994.

[6] P. Bosch and S. J. Mullender, "Real-Time Disk Scheduling in a Mixed-Media File System," *Proc. Real-Time Technology and Applications Symp.*, pp. 23-33, 2000.

[7] J. Bruno, E. Gabber, B. Ozden, and A. Silberschatz, "Disk Scheduling Algorithms with Quality of Service Guarantees," *Proc. IEEE Conf. Multimedia Computing Sys.*, June 1999.

[8] C. Chang, B. Moon, A. Acharya, C. Shock, A. Sussman, and J. Saltz. "Titan: a High-Performance Remote-Sensing Database," *Proc. the 13th Int'l Conf. Data Engineering*, Apr 1997.

[9] Y. Cho, M. Winslett, M. Subramaniam, Y. Chen, S. Kuo, and K. E. Seamons, "Exploiting Local Data in Parallel Array I/O on a Practical Network of Workstations," *Proc. the Fifth Workshop on I/O in Parallel and Distributed Systems*, pp.1-13, Nov. 1997.

[10] Jr. Coffman and M. Hofri, "Queueing Models of Secondary Storage Devices," Stochastic Analysis of Computer and Communication Systems, Ed. Hideaki Takagi, North-Holland, 1990.

[11] P. J. Denning, "Effects of Scheduling on File Memory Operations," *Proc. AFIPS Conf.*, 1967.

[12] Z. Dimitrijevic and R. Rangaswami, "Quality of Service Support for Real-time Storage Systems," *Proc. Int'l Conf. IPSI*, Sv. Stefan, Montenegro, October 2003.

[13] B. Forney, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, "Storage-Aware Caching: Revisiting Caching for Heterogeneous Storage Systems," *Proc. Int'l Symp. File and Storage Tech.*, 2001.

[14] B. Hendrickson and D. Womble, "The torus-wrap mapping for dense matrix calculations on massively parallel computers," *SIAM J. Sci. Comput.*, 15(5), Sept. 1994.

[15] J. Huber, C. L. Elford, D. A. Reed, A. A. Chien, and D. S. Blume, "PPFS: A high Performance Portable Parallel File System," *Proc. 9th ACM Int'l Conf. Supercomputing*, pp.385-394, July 1995.

[16] J. Hughes and D. Corcorna, "A Universal Access, Smart-Card-Based, Secure File System," *Atlanta Linux Showcase*, Oct. 1999.

[17] D. Jacobson and J. Wilkes, "Disk Scheduling Algorithms Based on Rotational Position," *Technical Report, HPL-CSP-91-7*, February 1991.

[18] W. B. Ligon and R. B. Ross, "Implementation and Performance of a Parallel File System for High Performance Distributed Applications," *Proc. IEEE Int'l Symp. High Performance Distributed Computing*, pp.471-480, August 1996.

[19] X. Ma, M. Winslett, J. Lee, and S. Yu, "Faster Collective Output through Active Buffering," *Proc. Int'l Symp. Parallel and Distributed Processing*, 2002.

[20] D. Mazieres, M. Kaminsky, M. Kaashoek and E. Witchel, "Separating key management from file system security," *Proc. ACM Symp. Operating System Principles*, December 1999.

[21] E. Miller, D. Long, W. Freeman and B. Reed, "Strong Security for Distributed File Systems," *Proc. Symp. File Systems and Storage Technologies*, January 2002.

[22] E. Nahum, S. O'Malley, H. Orman, R. Schroeppel, "Towards High Performance Cryptographic Software," *Proc. IEEE Workshop Arch. and Impl. of High Perf. Comm. Subsystems*, August 1995.

[23] K. W. Preslan, A. P. Barry, J. E. Brassow, G. M. Erickson, E. Nygaard, C. J. Sabol, S. R. Soltis, D. C. Teigland, and M. T. O'Keefe, "64-bit, Shared Disk File System for Linux," *Proc. NASA Goddard Conference on Mass Storage Systems*, March 1999.

[24] X. Qin, H. Jiang, Y. Zhu, and D. R. Swanson. "Improving the Performance of I/O-Intensive Applications on Clusters of Workstations" *Cluster Computing: The Journal of Networks, Software Tools and Applications*, Vol. 8, No. 4, Oct. 2005.

[25] A. Reddy and J. Wyllie, "Intergraded QoS Management for Disk I/O," *Proc. IEEE Conf. Multimedia Computing Sys.*, June 1999.

[26] R. Reist and S. Daniel, "A Continuum of Disk Scheduling Algorithms," *ACM Trans. on Computer Sys.*, pp.77-92, Feb. 1987.

[27] L. Reuther, M. Pohlack, "Rotational-Position-Aware Real-Time Disk Scheduling Using a Dynamic Active Subset," *Proc. IEEE Real-Time System Symp*, 2003.

[28] E. Riedel, M. Kallahalla, and R. Swaminathan, "A Framework for Evaluating Storage System Security," *Proc. the 1st Conf. File and Storage Technologies*, Monterey, CA, Jan. 2002.

[29] K. Salem and H. Garcia-Molina, "Disk Striping," *Proc. the 2nd Int'l Conf. Data Engineering*, pp.336-342, Feb. 1986.

[30] P. Scheuermann, G. Weikum, P. Zabback, "Data Partitioning and Load Balancing in Parallel Disk Systems," The VLDB Journal, pp. 48-66, July, 1998.

[31] M. Seltzer, P. Chen, and J. Ousterhout, "Disk Scheduling Revisited," *Proc. USENIX Technical Conf.*, pp.313-323, 1990.

[32] P. Shenoy and H. Vin, "Cello: A Disk Scheduling Framework for Next Generation Operating Systems," *Proc. ACM SigMetrics*, June 1998.

[33] T. Sumner and M. Marlino, "Digital libraries and educational practice: a case for new models," *Proc. ACM/IEEE Conf. Digital Libraries*, pp. 170 – 178, June 2004.

[34] T. Tanaka, "Configurations of the Solar Wind Flow and Magnetic Field around the Planets with no Magnetic field: Calculation by a new MHD," *J. Geophysical Research*, pp.17251-17262, Oct. 1993.

[35] T. Xie and X. Qin, "A New Allocation Scheme for Parallel Applications with Deadline and Security Constraints on Clusters," *Proc. IEEE Int'l Conf. Cluster Computing*, Boston, USA, Sept. 2005.

[36] T. Xie, X. Qin, and Andrew Sung, "SAREC: A Security-Aware Scheduling Strategy for Real-Time Applications on Clusters," *Proc. 34th Int'l Conf. Parallel Processing*, Norway, June 2005.

[37] T. Xie and X. Qin, "Enhancing Security of Real-Time Applications on Grids through Dynamic Scheduling," *Proc. 11th Workshop Job Scheduling Strategies for Parallel Processing*, June 2005.

[38] P. S. Yu, M. S. Chen, and D. D. Kandlur, "Grouped Sweeping Scheduling for DASD-Based Multimedia Storage Management," *ACM Multimedia Systems*, Vol.1, No.3, pp.99-109, 1993.