This paper appeared in *Proc. IEEE/ACM 6th Int'l Symp. Cluster Computing and the Grid* (CCGrid), 2nd Int'l Workshop on Cluster Security, May 2006.

SAHA: A Scheduling Algorithm for Security-Sensitive Jobs on Data Grids

Tao Xie and Xiao Qin*
Department of Computer Science
New Mexico Institute of Mining and Technology
801 Leroy Place, Socorro, New Mexico 87801-4796
{xietao, xqin}@cs.nmt.edu

Abstract

Security-sensitive applications that access and generate large data sets are emerging in various areas such as bioinformatics and high energy physics. Data grids provide data-intensive applications with a large virtual storage framework with unlimited power. However, conventional scheduling algorithms for data grids are inadequate to meet the security needs of data-intensive applications. To remedy this deficiency, we address in this paper the problem of scheduling data-intensive jobs on data grids subject to security constraints. Using a security- and data-aware technique, SAHA (Security-Aware and Heterogeneity-Aware scheduling strategy) is proposed to *improve quality of security for data-intensive applications running on data grids. Results based on real-world traces show that the proposed scheduling scheme dramatically improves security and performance over two existing scheduling algorithms.*

1. Introduction

A grid is a collection of geographically dispersed computing resources, providing a large virtual computing system to users [3]. There are four commonly used kinds of resources in grids: computation, storage, software, and communications. Data grids have been developed to integrate heterogeneous data archives stored in a large number of geographically distributed sites into a single virtual data management system [2]. Furthermore, data grids provide diverse services to fit the needs of high-performance and data-intensive computing [7].

There have been some efforts to develop data-intensive applications such as bioinformatics and high energy physics in data grid environments [8][11]. In the past few years, many studies have addressed the issue of scheduling for data grids [6][8]. Although existing scheduling algorithms provide high performance for data-intensive applications, these algorithms are not adequate for data-intensive applications that are security-sensitive in nature.

Security services are critically important to a variety of security-sensitive applications on grids in general [10] and data grids in particular. This is because sensitive data and processing require special safeguard and protection against unauthorized access. Much attention has been focused on security for applications running on grids [4][5][10]. Our proposed scheduling scheme for security-sensitive jobs is complementary to existing security techniques for grids in the sense that an additional improvement in security can be achieved by combining our strategy with the existing grid security services.

Very recently, Song *et al.* proposed security-driven scheduling algorithms for grids [9]. We proposed an array of security-aware scheduling algorithms for applications with timing constraints on single machines [14], clusters [12], and grids [13]. The above algorithms can improve security and performance of various applications. However, these algorithms have no inherent capability of supporting data girds due to ignorance of heterogeneous resources and data sets read/write by applications.

In data grids, a growing number of applications have both security and data constraints. In this regard, we are motivated to introduce a new concept, namely degree of security deficiency. Additionally, a scheduling strategy is proposed to enhance security of data-intensive applications on data grids.

The main contributions of this study include: (1) a model for data-intensive applications with security requirements; (2) a concept of degree of security deficiency; (3) considerations of datasets accessed and generated by applications; (4) integration of security heterogeneity into job scheduling; (5) a scheduling strategy for data-intensive applications on data grids.

The rest of the paper is organized in the following way. Section 2 presents a system model used to construct the scheduling strategy. A security overhead model is outlined in Section 3. The scheduling strategy is detailed in Section

* Contact author. http://www.cs.nmt.edu/~xqin

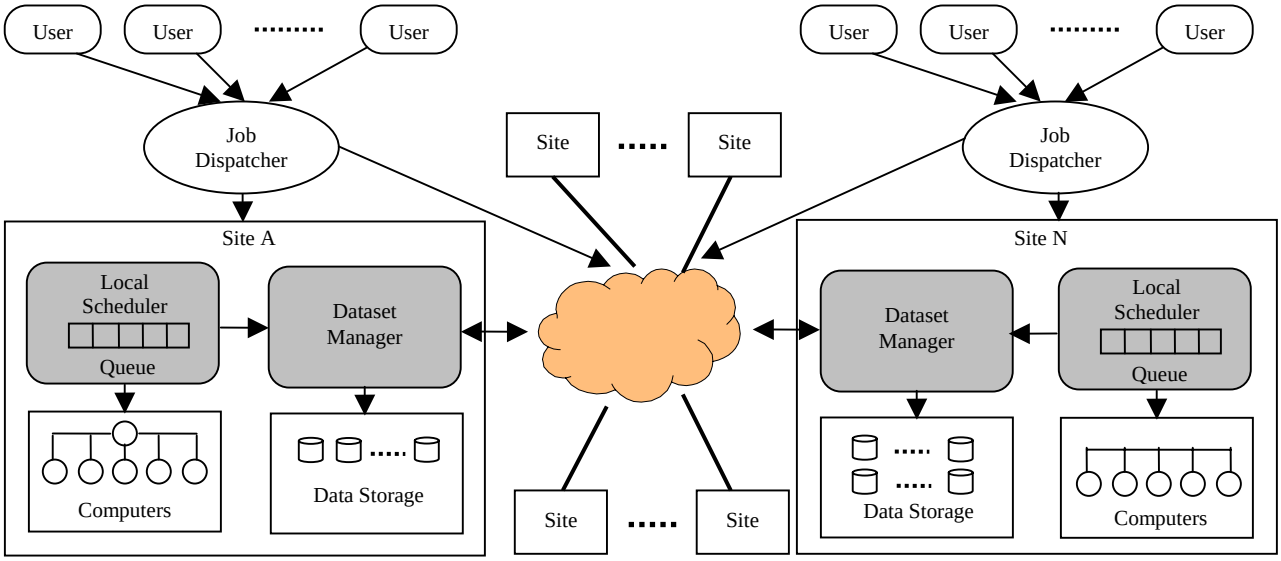

Figure 1. System Model of a Data Grid

4. Section 5 discusses experimental results that confirm the validity of the proposed scheduling strategy. Section 6 concludes the paper with summary and future research directions.

2. System and Job Models

We model a data grid as a network of n heterogeneous sites with various system architectures. Each site in the data grid is comprised of data stores and computational facilities. Let $M = \{M_1, M_2, \ldots, M_n\}$ denote a set of sites, each of which is represented a pair $M_i = (N_i, D_i, P_i)$, where N_i is a set of nodes residing in the site, D_i is an array of data sets accessed and generated by data-intensive jobs, $P_j = (p_j^1, p_j^2, \ldots, p_j^q)$ is a vector characterizing the security levels provided by the site.

Fig. 1 shows the system model of a data grid. Each site has a job dispatcher that accommodates arrival jobs submitted by multiple users. If a job is allocated to a remote site, it will be dispatched to the other site through a high-speed network. Otherwise, the job will be placed into a local queue managed by a local scheduler. The complete functionality of dataset manager is three-fold. (1) It monitors local datasets. If popularity of a dataset exceeds a threshold value, the dataset manager will automatically replicate the dataset to a remote site. (2) It accepts dataset requests issued by the local scheduler. When a job needs a remote dataset to run, the local scheduler passes a dataset request to the dataset manager. (3) If a dataset request is sent from the local scheduler, the dataset manager will deliver the request to a destination site.

Each data-intensive job considered in this study consists of a collection of tasks performed to accomplish an overall mission. Without loss of generality, we assume that tasks in a job are independent of one another. A job is modeled as a set of parameters, e.g., $J = (a, u, E, c, d, L, SC, SD)$, where a is the arrival time, u is the number of requested nodes, c is the size of the job code, d denotes the size of input/output data set, and L represents a group of sites where the data set is stored. E is a vector of execution times for the job on each site in M, and $E_i = (e_i^1, e_i^2, \ldots, e_i^n)$. The security requirement of job J is modeled by a vector of security levels, e.g., $SC = (sc^1, sc^2, \ldots, sc^q)$, where q is the number of required security services. sc^k is the required security level of the kth security service. Similarly, the security requirement of the job's data set is modeled by a vector of security levels, e.g., $SD = (sd^1, sd^2, \ldots, sd^q)$.

3. Security Overhead Model

We build in this section a security overhead model for data-intensive jobs running on a data grid, where input data sets are replicated in data grid environment.

First, we consider a case where input data is locally available and processing is performed on local site. Let sc_i^k and $c_j^k(sc_i^k)$ be the security level and overhead of the kth security service for job J_i. Likewise, let sd_i^k and $c_j^k(sd_i^k)$ denote the security level and overhead of the kth security service for the job's data set. The security overhead c_{ij} experienced by job J_i on local site M_i where the data set is available can be computed as a sum of times spent in securing the application code and data set. Thus, we have the following equation, where $sc^k \in SC$ and

$sd^k \in SD$.

$$c_{ij} = \sum_{k=1}^{q} c_j^k(sc_i^k) + \sum_{k=1}^{q} c_j^k(sd_i^k),$$
$$= \sum_{k=1}^{q} \left[c_j^k(sc_i^k) + c_j^k(sd_i^k) \right], \quad (1)$$

Second, we derive an expression to calculate the security overhead of a locally executed job J_i accessing its remote data set. In this scenario, job's data set needs to be fetched from a remote site through networks. Suppose there are *p* number of security services provided for a link between site *v* and site *j*. The security overhead of the *k*th security service for the data set delivered from site *v* to *j* is expressed as $c_{vj}^k(sd_i^k)$. The total security overhead is the sum of the security overhead caused by data transfer, application code, and data set protections. Therefore, the security overhead in this case can be written as:

$$c_{ij} = \sum_{k=1}^{p} c_{vj}^k(sd_i^k) + \sum_{k=1}^{q} \left[c_j^k(sc_i^k) + c_j^k(sd_i^k) \right], \quad (2)$$

Third, Expression (3) is used to compute the security overhead of a remotely executed job J_i that accesses its data set on a local site. Thus, the application code needs to be transmitted to the remote site where the data is stored. The security overhead of the *k*th security service for transmitting the application code from remote site *v* to local site *j* is denoted by $c_{vj}^k(sc_i^k)$. Finally, the total security overhead in this case can be calculated as:

$$c_{ij} = \sum_{k=1}^{p} c_{vj}^k(sc_i^k) + \sum_{k=1}^{q} \left[c_j^k(sc_i^k) + c_j^k(sd_i^k) \right], \quad (3)$$

where the two terms on the right-hand side of Equation (3) are the security overhead incurred in securing the application code transfer, application code, and data set.

Last, we consider the security overhead of a job executed on a remote site to which the job's data set is moved. In this scenario, the security overhead includes two major categories of cost: (a) overhead of security services for transmitting the application code and data set; (b) time spent in securing the job's code and data set. Thus we have the following equation:

$$c_{ij} = \sum_{k=1}^{p} \left[c_{vj}^k(sc_i^k) + c_{vj}^k(sd_i^k) \right] + \sum_{k=1}^{q} \left[c_j^k(sc_i^k) + c_j^k(sd_i^k) \right] \quad (4)$$

4. Scheduling Strategy

4.1 Degree of Security Deficiency

Before proceeding to the design of the scheduling strategy, we need to construct a model to measure quality of security provided by a data grid. Specifically, we introduce a new concept of security deficiency quantified as the discrepancy between requested security levels and offered security levels. Thus, the security deficiency of the *k*th security service for job J_i on site N_j is:

$$\delta_{site}(sc_i^k, sd_i^k, p_j^k) = \begin{cases} 0, \text{if } sc_i^k \le p_j^k \text{ and } sd_i^k \le p_j^k \\ sc_i^k - p_j^k, \text{if } sc_i^k > p_j^k \text{ and } sd_i^k \le p_j^k \\ sd_i^k - p_j^k, \text{if } sc_i^k \le p_j^k \text{ and } sd_i^k > p_j^k \\ sc_i^k + sd_i^k - 2p_j^k, \text{otherwise} \end{cases}. \quad (5)$$

With regard to the *k*th security service, the security deficiency of a data set delivered from site *v* to *j* is defined by the following equation.

$$\delta_{link}(sc_i^k, p_{vj}^k) = \begin{cases} 0, \text{if } sc_i^k \le p_{vj}^k \\ sc_i^k - p_{vj}^k, \text{otherwise} \end{cases}, \quad (6)$$

where p_{vj}^k is the security levels provided by the link.

Similarly, the security deficiency for application code transmitted from site *v* to site *j* is expressed as:

$$\delta_{link}(sd_i^k, p_{vj}^k) = \begin{cases} 0, \text{if } sd_i^k \le p_{vj}^k \\ sd_i^k - p_{vj}^k, \text{otherwise} \end{cases}, \quad (7)$$

A small security deficiency indicates a high degree of service satisfaction. The security deficiency of a job J_i on site M_j can be derived from the following Equation.

$$SD_{ij} = \sum_{k=1}^{q} \left[w_i^k \delta_{site}(sc_i^k, sc_i^k, p_j^k) \right] + \sum_{v=1}^{n} \left[xc_{vj} \sum_{k=1}^{p} \left[w_i^k \delta_{link}(sc_i^k, p_{vj}^k) \right] \right] + \sum_{v=1}^{n} \left[xd_{vj} \sum_{k=1}^{p} \left[w_i^k \delta_{link}(sd_i^k, p_{vj}^k) \right] \right], \quad (8)$$

where xc_{vj} and xd_{vj} are two step functions. xc_{vj} = 1 if the application code is moved from site *v* to *j*, otherwise xc_{vj} = 0. xd_{vj} = 1 if the dataset is fetched from site *v* to *j*, otherwise xd_{vj} = 0. w_i^k is the weight of the *k*th security

service, e.g., $0 \le w_i^k \le 1$, and $\sum_{k=1}^{q} w_i^k = 1$. Note that weights reflect relative priorities given to security services.

Given a sequent of data-intensive jobs *J*, The *degree of security deficiency* for a data grid *M* under allocation *X* is defined as the sum of the security deficiencies of all the jobs. The degree of security deficiency can be written as:

$$DSD(M, J, X) = \sum_{\forall J_i} \sum_{j=1}^{n} \left[x_{ij} SD_{ij} \right] \quad (9)$$

where $x_{ij} \in X, x_{ij} = 1$ if J_i is allocated to site M_j, $\sum_{j=1}^{n} x_{ij} = 1$.

1.**for** each job J_i submitted to site M_j **do**
2. **for** each site M_k in the system **do**
3. Use **Equation (9)** to obtain degree of security deficiency;
4. Use **Equation (10)** to compute the earliest start time;
5. Use **Equations (1)-(4)** to compute security overhead;
6. Computer the finish time of J_i on site M_j
7. **end for**
8. Sort all sites in earliest finish time;
9. **for** a group sites providing the minimal finish time **do**
10. Find site M_t that minimize the degree of security deficiency;
11. Update J_i's start and finish times, and schedule information;
12. Dispatch J_i and its dataset to site M_t for execution;
13.**end for**

Figure 2. The SAHA Strategy

4.2 Scheduling Strategy Description

Given a data grid *M* and a sequence of jobs *J*, our scheduling strategy is intended to generate an allocation *X* minimizing the degree of security deficiency computed by Equation (9). Finally, we can obtain the following non-linear optimization problem formulation:

$$\text{minimize } DSD(M, J, X) = \sum_{\forall J_i} \sum_{j=1}^{n} \left[x_{ij} SD_{ij} \right]$$

$$\text{subject to } J_i \in J, \sum_{j=1}^{n} x_{ij} = 1.$$

Now we present the Security-Aware and Heterogeneity-Aware scheduling strategy (SAHA). The earliest start time of job j_i on site M_j can be approximated as below:

$$\text{es}_j(J_i) = r_j + \sum_{J_l \in M_j} \left(e_l^j + \sum_{k=1}^{q} \left[c_{lj}^k (sc_l^k) + c_{lj}^k (sd_l^k) \right] \right), \quad (10)$$

where r_j represents the finish time of a job currently running on the *j*th site, and the second term on the right-hand side is the overall execution time (security overhead is factored in) of waiting jobs assigned to site M_j prior to the arrival of J_i. In other words, the earliest start time of J_i is the sum of the finish time of the running task and the overall execution times of the jobs with earlier arrival on site M_j.

The SAHA strategy is outlined in Fig. 2. The goal is to maximize security while maintaining high performance for data-intensive jobs running on data grids. In order to improve security, SAHA makes an effort to minimize degree of security deficiency measured by Equation 9 (see Steps 9-10 in Fig. 2) without performance deterioration.

Before optimizing the degree of security deficiency, step 3 manages to calculate the degree of security deficiency for the submitted job on site M_j. SAHA sorts all the sites in a non-decrease order of earliest finish times estimated by Equation 10 (see Steps 4 and 8). From the group of sites providing the minimal finish times, step 10 chooses the most appropriate site that substantially reduces the degree of security deficiency. Step 11 is intended to update information pertinent to scheduling decisions, whereas step 12 dispatches the job and its dataset to the selected site for execution.

4.3 Risk-Free Probability

We derive in this subsection the probability $P_{rf}(J_i, M_j)$ that J_i remains risk-free during the course of its execution.

The quality of security of a task with respect to the *k*th security service is calculated as $\exp\left(-\lambda_i^k \left(e_i^j + c_{ij}\right)\right)$ where λ_i^k is the task's risk rate of the *k*th security service, and c_{ij} is the security overhead experienced by the job on site *j*. The risk rate is expressed as:

$$\lambda_i^k = 1 - \exp\left(-\alpha(1 - s_i^k)\right) \quad (11)$$

The quality of security of J_i on site M_j can be obtained below by considering all security services.

$$P_{rf}(J_i, M_j) = \prod_{k=1}^{q} \exp(-\lambda_i^k \left(e_i^j + c_{ij}\right)) = \exp\left(-\left(e_i^j + c_{ij}\right) \sum_{k=1}^{q} \lambda_i^k \right) \quad (12)$$

Finally, we obtain the overall quality of security of task J_i in the system as follows,

$$P_{rf}(J_i) = \sum_{j=1}^{n} \left\{ p_{ij} \cdot P_{rf}(J_i, M_j) \right\}, \quad (13)$$

where p_{ij} is the probability that J_i is allocated to site M_j.

Table 1. Characteristics of System Parameters

Parameter	Value (Fixed) - (Varied)
Number of jobs	(6400) –
Number of sites	(32) – (32, 64,128)
Number of datasets	(200) – (100, 200, 400)
Job arrival rate	Decided by the trace
Network bandwidth	1 Gbps (billions of bits/Sec.)
Data to be secured	1–10MB short, 10-50MB medium, 50-100 MB long
Size of datasets	(500-800 MB short jobs, 800MB-1GB medium jobs, 1-2GB long jobs) –
Number of sites for each dataset	(1) – (1, 4, 8)
Size of site group	(8) – (1, 2, 4, 8, 16)
Site security level	(0.1 – 1.0)
Job security level	(0.1 – 1.0)
Required security services	Encryption, Integrity and Authentication
Weights security services	Authentication weight=0.2, Encryption weight=0.5, Integrity weight=0.3

5. Experimental Results

We use simulation experiments based on San Diego Supercomputer Center (SDSC) SP2 traces to evaluate benefits of the SAHA strategy. To demonstrate the strengths of SAHA, we compared it with two well-known data grid scheduling algorithms, namely, *JobRandom* and *JobDataPresent* [8], among which JobDataPresent is the best algorithm based on results reported in [8]. The two algorithms are briefly described as below. (1) *JobRandom*: For each submitted job, a randomly selected site is allocated to the job. (2) *JobDataPresent*: For each submitted job, a site that has required datasets is assigned to the job. Table 1 summarizes system parameters of the simulated data grid used in our experiments.

5.1 Simulator and Simulation Parameters

Dataset popularities are randomly generated with a regional uniform distribution. All submitted jobs in the trace fall into three categories based on their execution times. The categories include short jobs, medium jobs, and long jobs. Accordingly, we assign each category a dataset size range (region). Dataset popularities within each range are uniformly distributed.

We modified the trace by adding a block of data to be secured for each job in the traces. The size of the security-sensitive data assigned to each job is generated according to a uniform distribution. The performance metrics we used include: *degree of security deficiency* (see Equation 1), *risk-free probability* (see Equation 9), *Average Data Transferred* (defined as the volume of transferred data per job), *average response time* (the response time of a job is the time period between the job's arrival and its completion and the average response time is the average value of all jobs' response time), *slowdown ratio* (the slowdown of a job is the ratio of the job's response time to its service time and the slowdown ratio is the average value of all jobs' slowdowns), *Average idle time percentage* (an idle time percentage of a site is the ratio of

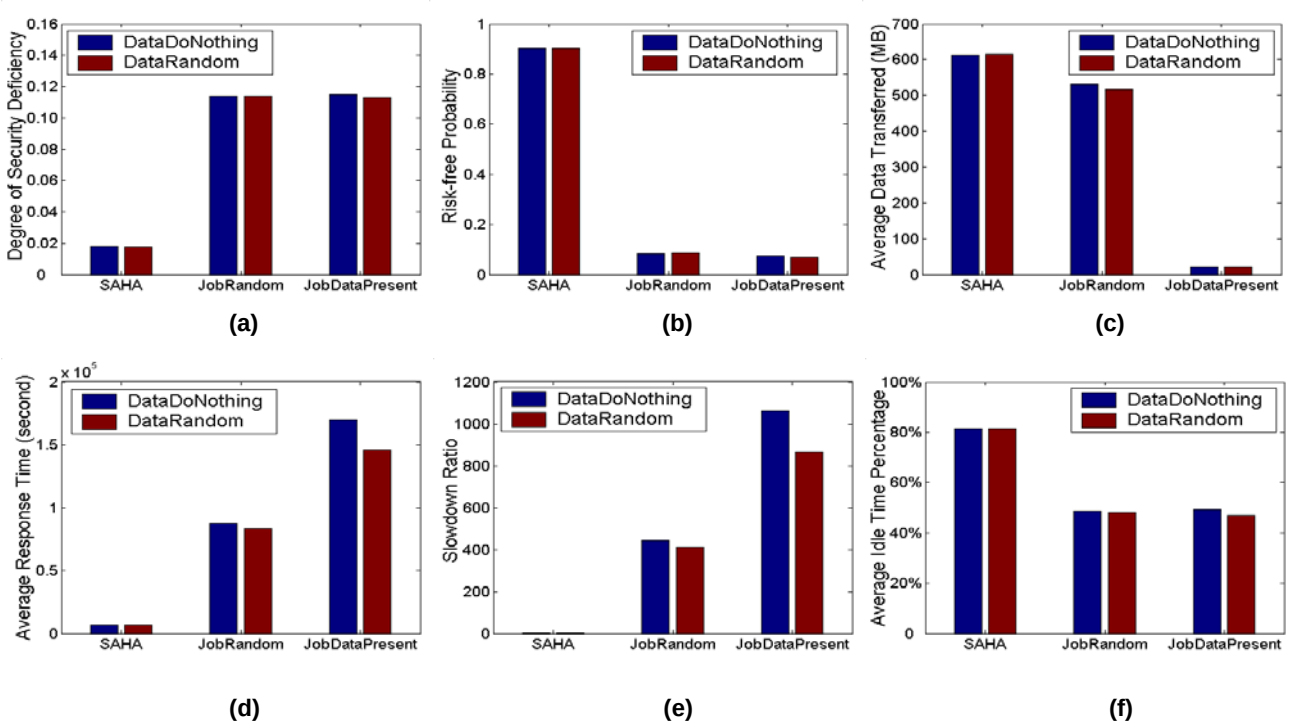

Figure 3. Overall performance comparisons.

the site's idle time to its last job finish time).

5.2 Overall Performance Comparisons

The goal of this experiment is to compare the proposed SAHA algorithm against the two heuristics in two dataset replication methods, *DataDoNothing* and *DataRandom*.

Fig. 3 shows the simulation results for the three scheduling algorithms on a data Grid with 32 sites. We observe from Fig. 3 that SAHA significantly outperforms the two heuristics in terms of degree of security deficiency (Fig. 3a) and risk-free probability (Fig. 3b), whereas JobRandom and JobDataPresent algorithms exhibit similar performance. We attribute the performance improvement to the fact that SAHA is a security-sensitive scheduler and judiciously assigns a job to a site not only considering its computational time but also its security demands. In addition, SAHA is greatly superior to the two alternatives in conventional performance metrics such as average response time (Fig. 3d) and slowdown ratio (Fig. 3e), with which users are mostly concerned. The performance improvements of SAHA are at the cost of a higher volume of data transferred. However, the data transmission overhead can be largely alleviated by using a high network bandwidth (see Table 1) that is available in the real world. Also, SAHA has a higher average idle time percentage (Fig. 3f) because it can generate an effective schedule that leads to a much shorter average execution time for the job set. The higher idle time percentage suggests that the Grid has potential to accommodate more jobs when the SAHA scheduling algorithm is in place.

6. Summary and Future Work

In this paper, a novel security- and heterogeneity-driven scheduling strategy was proposed for data-intensive applications on data grids. In the first part of this study, we constructed a model for data-intensive jobs with security requirements. In addition, we built a new security overhead model to measure security overheads experienced by data-intensive jobs. Our scheduling strategy takes into account datasets accessed and generated by applications, thereby maintaining high performance for data grids. By using the new concept of degree of security deficiency, this scheduling strategy is capable of improving quality of security for data grids. We implemented a trace-driven simulator to evaluate the performance of our approach under a wide range of workload characteristics. Compared with two existing scheduling algorithms, our strategy achieved improvements in security and performance by up to 810% and 1478%, respectively.

We qualitatively assigned security levels to the security services based on their respective security capacities. In our future work, we intend to investigate a quantitative way of reasonably specifying the security level of each security mechanism.

Acknowledgements

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] T.F. Abdelzaher and K.G. Shin., "Combined Task and Message Scheduling in Distributed Real-Time Systems," *IEEE Trans. Parallel and Distributed Systems*, Vol. 10, No. 11, 1999.

[2] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets, " *J. of Network and computer Appl.*, Vol. 23, 2001.

[3] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *Int'l Journal of Supercomputer Applications*, Vol. 15, No. 3, 2001.

[4] K. Keahey and V. Welch, "Fine-Grain Authorization for Resource Management in the Grid Environment," *Proc. Int'l Workshop Grid Computing*, 2002.

[5] J. Novotny, S. Tuecke, and V. Welch, "An Online Credential Repository for the Grid: MyProxy," *Proc. Int'l Symp. High Performance Distributed Computing*, August 2001.

[6] S.-M. Park and J.-H Kim, "Chameleon: A Resource Scheduler in a Data Grid Environment," *Proc. Int'l Symp. Cluster Computing and the Grid*, 2003.

[7] X. Qin and H. Jiang, "Data Grids: Supporting Data-Intensive Applications in Wide Area Networks," *High Performance Computing: Paradigm and Infrastructure*, edited by L. Yang and M.-Y Guo, John Wiley and Sons, 2004.

[8] K. Ranganathan and I. Foster, "Decoupling Computation and Data Scheduling in Distributed Data-Intensive Applications," *Proc. IEEE Int'l Symp. High Performance Distributed Computing*, pp.352-358, 2002.

[9] S. Song, Y.-K. Kwok, and K. Hwang, "Trusted Job Scheduling in Open Computational Grids: Security-Driven Heuristics and A Fast Genetic Algorithms," *Proc. Int'l Symp. Parallel and Distributed Processing*, 2005.

[10] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, S. Tuecke, "Security for Grid Services," Proc. *Int'l Symp. High Performance Distr. Computing*, 2003.

[11] L. Winton, "Data Grids and High Energy Physics: A Melbourne Perspective," *Space Science Reviews*, 107 (1-2): pp.523-540, 2003.

[12] 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.

[13] 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*, MA, June 2005.

[14] T. Xie, A. Sung, and X. Qin, "Dynamic Task Scheduling with Security Awareness in Real-Time Systems", *Proc. Int'l Symp. Parallel and Distributed Processing*, April 2005.