Security-Driven Scheduling Model for Computational Grid using Genetic Algorithm

R. Kashyap, D.P. Vidyarthi

Abstract— **The conflict between achieving good performance in terms of time etc. and achieving high quality of security protection introduces new challenges in security critical grid scheduling. Extensive study indicates that the scheduling performance is affected by the heterogeneities of security and computational power of resources. Different tasks may have varied security requirement and even for the same security requirement, security overhead may vary for different node at which the task is scheduled. In this paper, a security driven scheduling using genetic algorithm (SDSG) is proposed which aims at maximizing security while restricting security overhead under a certain limit. Extensive simulation results over dynamically created heterogeneous grid environment reveal that SDSG achieves better security and exhibit less security overhead and makespan in comparison to other such algorithms viz. MinMin MaxMin, SPMinMin and SPMaxMin.**

Index Terms— **Grid computing, Security-aware grid scheduling, Cipher suite, Makespan, Response time**

I. Introduction

A computational grid is a collection of geographically dispersed heterogeneous computing resources, giving the image of a single large virtual computing system to users [1][2][3]. Scheduling on such platform is an important aspect more so, being this a heterogeneous system. The main challenge of task scheduling in grids is its highly dynamic environment, where the computing resources have their own access policies, security, availability etc. At the same time, resources are of greater heterogeneity ranging from desktop PCs to supercomputers. Grid computing has often extensively supported collaborative projects on the internet. Most of these projects have stringent security requirements.

The application itself imbibes security up to some extent, but more usually it is to be supported and ensured by the grid environment. The dynamic and multi-institutional nature of the grid introduces challenging security threats warranting the development of the new technical approaches towards this problem. In a security aware environment, responsibility is delegated to the scheduler for allocating the task on those resources that gives best possible security, while understanding the computational and security heterogeneity of the resources.

R. Kashyap is with Lal Bahadur Shastri Institute of management, Delhi, India (phone: 91-11-25307700; fax: 91-11-24522474; e-mail: rekhakashyap@lbsim.ac.in).

D. P. Vidyarthi is with School of Computer Science, Jawaharlal Nehru University, Delhi, India (dpv@jnu.ac.in).

Task scheduling in grid is an *NP*-hard optimization problem, so many heuristic and meta-heuristics algorithms are in use towards finding suboptimal solutions, i.e., solutions whose optimality cannot be guaranteed. Meta-heuristics like Simulated Annealing (SA) [4], Genetic Algorithm (GA) [5], Ant Colony Optimization (ACO) [6], particle Swarm Optimization (PSO) [7], etc. are also used for grid scheduling as they generally produce higher quality results than simple heuristics, although may take a bit longer as they have to generate and evaluate many solutions rather than just one. These nature based meta-heuristics follow the Darwin's natural selection law i.e. only the fittest can survive. GA a population-based meta-heuristic, was created by John Holland [5] and produces the next generation with the techniques inspired by evolutionary biology, such as inheritance, mutation, crossover, and selection. GA considers a solution as an organism, thus better the quality of the solution higher is the survival probability, through crossover (also called recombination) and mutation. GA can escape from the local optimal to search for the global optimal. In this paper, we propose a genetic algorithm for job scheduling to address the heterogeneity of security mechanism in a computational grid. The proposed Security Driven Scheduling using Genetic algorithm (SDSG) improves the security of the heterogeneous grid while restricting the security overhead within a limiting range.

Next section discusses some related work in this field. Scheduling strategy is described in section 3. Section 4 briefs the security model used in this work. Proposed SDSG is analyzed in Section 5. Experimental results and observations for SDSG and the compared heuristics are presented in Section 6 while making the conclusion in the last Section.

II. Related Work

To achieve the promising potential of underlying distributed resources in the grid, effective scheduling algorithms are fundamentally important. Scheduling, an NP-hard problem, is rather complex one as the Grid being a geographically dispersed heterogeneous multiprocessing environment. Consequent to this is the emergence of many heuristic and evolutionary approaches towards this problem [8] [9] [10] [11] [12] [13]. Some well known heuristic based grid scheduling algorithms, proposed in the literature, are as follows. Casanova et al. [14] proposed an adaptive grid scheduling algorithm for parameter sweep applications, where tasks can share input files and also extended Sufferage heuristics as XSufferage. DFPLTF (Dynamic

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Fastest Processor to Largest Task First) is a scheduling heuristic, which gives highest priority to the largest task [15]. Fujimoto and Hagihara [16] have proposed Round Robin (RR) grid scheduling algorithm for parameter sweep applications. MinMin and MaxMin are well known algorithms used in real world distributed resource management systems such as SmartNet [17]. MinMin gives highest priority to the task that can be completed first. In MinMin, the grid site offering the earliest completion time is tagged and the task that has the minimum earliest completion time is allocated to the respective site. MaxMin also tags the grid site that offers the earliest completion time but highest priority is given to the task that has maximum earliest completion time. All the above mentioned algorithms are not security aware and hence unsuitable for security aware applications.

The goal of a security aware scheduler is to meet the desired security requirements and at the same time offer a high level of performance with respect to one or more parameters e.g. makespan, average response time, site utilization etc. [18][19][20]. Further, security heterogeneity and the grid dynamism makes security aware grid scheduling more challenging as the security overhead is node dependent. Some of the security-aware schedulers discussed in the literature are as follows. Song, Kwok and Hwang [21] envision a secured scheduling framework with the risk involved while dispatching the jobs to the remote nodes. They proposed three scheduling strategies based on different risk levels and modified the MinMin and Sufferage heuristics in three modes; a) Secure mode (jobs were only scheduled to those nodes which can ensure security) b) Risky mode (jobs were scheduled to any available nodes without considering the risks between jobs and nodes), and c) F-risky mode (jobs were scheduled to available nodes to take at most F risks). SPMinMin and SPMaxMin [22] [23] are improvement over the Secure mode suggested by Song. In SPMinMin and SPMaxMin security requirement is the guiding parameter for scheduling decision and they guarantee the security of the job while minimizing the makespan. SATS, suggested by Xie and Qin [24], takes into account heterogeneities in security and computation. SATS also provides a means of measuring overhead incurred by security services and tries to improve security and minimize makespan. Xiaoyong et al. [25] incorporated security into inter-task dependency and proposed a security driven scheduling algorithm (SDS), to improve security of HDS (Heterogeneous distributed systems) while minimizing the makespan, risk probability and speedup. Our work is different from SATS and SDS as they have proposed heuristics that didn't search the whole range of search space. Proposed work which is based on GA can escape from the local optimum to search for the global optimum. Chao-Chin and Ren-Yi [26] proposed a genetic algorithm, addressing the heterogeneities of fault tolerant mechanism in computational grid. They improved on job failure rate optimizing the makespan, whereas the proposed algorithm improves the total security value minimizing the security overhead.

III. SCHEDULING STRATEGY

The proposed model considers a grid consisting of number of non dedicated processing nodes which, in turn, may have a single processor or a group of heterogeneous or homogeneous processors. A job is comprised of *"n"* independent tasks with different computational size and security level. The tasks have soft deadlines and are independent i.e. without any precedence constraint. The list of terminologies, used in this paper, is as follows.

A task T_i is characterized as $T_i = (Sz_i, SL_i)$ where, Sz_i is the computational size in millions of instructions and SL_i and the security level assigned to the i^{th} task.

Processing node N_j of the grid is characterized as $N_j = (SP_j, BT_j)$ where, SP_j is the speed of node in MIPS and BT_j is the begin time of the node (time to execute already assigned tasks to the node).

A schedule of the job is depicted as the set of n tuples $<T_i, P_j, BT_j, SO_{ij}, CT_{ij}>$ in which, T_i is i^{th} task, P_j is j^{th} processing node, BT_j is Begin Time at j^{th} processing node, SO_{ij} is the Security Overhead of i^{th} task on j^{th} node and CT_{ij} is the completion time of the i^{th} task on the j^{th} processing node. CT_{ij} is calculated using equation 1.

$$CT_{ij} = BT_j + ET_{ij} + SO_{ij} \qquad (1)$$

where, ET_{ij} is Execution Time of i^{th} task on j^{th} processing node and SO_{ij} is the Security Overhead of i^{th} task on j^{th} node. Begin time of every node at the start of schedule is assumed to be zero but once the execution start, it will be affected.

IV. SECURITY STRATEGY

A. Security Model

One of the key factors behind the growing interest in grid computing is the evolution of standards such as TCP/IP and Ethernet in networking. For the TCP networking model IPsec, TLS/SSL and SSH are the popularly used security protocols operating on its network, transport and application layer respectively [27] [28] [29][30]. These protocols offer security to any grid application by the common security services of key exchange, authentication, confidentiality and integrity. Each protocol is further configured to match differing security requirements through cipher suites negotiations where cipher suite, is a named combination of key exchange, authentication, encryption, and integrity algorithms used to negotiate the security settings for a network connection. In the present work, SSL V3 protocol is considered and security levels are assigned for the cipher suites supported by it. *SL* for each cipher-suite is based on the weighted sum of security services involved in the cipher-suite. Cipher-suite offering more security (algorithms with longer keys) has more computational cost and therefore is assigned a higher security level. The security level also provides a mechanism for calculating the security overhead expenses. Subset of cipher suites supported by SSL V3 protocol is shown in Table 1. The third row SSLCipherSpec SSL_RSA_WITH_DES_CBC_SHA indicates use of DES with 56-bit encryption. The fourth row SSL CipherSpec SSL_RSA_WITH_3DES_ EDE_ CBC_SHA indicates use

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

of 3DES with 168-bit encryption. Among the six cipher suites, mentioned in the Table 1, the first one provides the weakest security and the last one provides the strongest security.

TABLE 1
THE SUBSET OF CIPHER SUITES SUPPORTED BY SSL V3 PROTOCOL

SSLCipherSpec	SSL_RSA_WITH_RC4_128_MD5	Security Level 1
SSLCipherSpec	SSL_RSA_WITH_RC4_128_SHA	Security Level 2
SSLCipherSpec	SSL_RSA_WITH_DES_CBC_SHA	Security Level 3
SSLCipherSpec	SSL_RSA_WITH_3DES_EDE_CBC_SHA	Security Level 4
SSLCipherSpec	SSL_RSA_EXPORT_WITH_RC4_40_MD5	Security Level 5
SSLCipherSpec	SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5	Security Level 6

B. Security Overhead Computation

Security overhead is calculated as suggested by Tao Xie and Xiao Qin [31] (equation 2), where, SL_i is in the range [1, 2, 3…..R] and 1 and R are the lowest and highest security level. For the experiments, shown in this paper, R is set to be 15. The rationale behind this security overhead model is based on the observation that security overhead of a particular application tends to be proportional to the execution time of the application. In other words, security overhead depends upon the amount of data to be secured and thus is the product of the execution time (which depends upon data size) and relative security required as shown in equation 2. Xie Tao [24] and Xiaoyong Tang [25] have proposed more precise model for calculating security overhead in which they calculate security overhead for each security service namely authentication integrity and confidentiality. Simpler security overhead calculation has been effectuated in all the algorithms since for the comparison purposes; the result will not be affected. Total computation time considering security overhead is shown in equation 3.

$$SO_{ij} = ET_{ij}\left(SL_i / R\right) \qquad (2)$$

$$CT_{ij} = BT_j + ET_{ij}\ (1 + SL_i / R) \qquad (3)$$

V. THE PROPOSED WORK

The paper proposes a security driven scheduling strategy using Genetic algorithm.

A. Security Driven Scheduling using Genetic Algorithm (SDSG)

The aim of the proposed algorithm is to get maximum security benefit while minimizing the security overhead. An important decision in such multiobjective optimization is how to evaluate the quality of solutions since the conflicting and incommensurable nature of some of the criteria makes this process more complicated. The possible alternatives are as follows.

Combine the objectives: This is one of the classical methods to evaluate the solution fitness in multiobjective optimization. It refers to converting the multiobjective problem into a single-objective one by combining the various criteria into a single scalar value. The most common way of doing this is by setting weights to each criterion and add them all together using an aggregating function.

*Pareto-based evaluation***:** In this approach, a vector containing all the objective values represents the solution fitness and the concept of dominance is used to establish preference between solutions. A solution *x* is said to be non inferior or non-dominated if there is no other solution that is better that *x* in all the criteria.

Alternating the objective: This is also an approach that has been used for many years. It refers to optimizing one criterion at a time while imposing constraints on the others. We obtain the fitness function by optimizing security value while keeping a limit on the security overhead. The constrained value of security overhead is obtained by averaging the security overhead, from randomly created 500 schedules.

B. Coding of Solutions

The encoding of individuals (also known as chromosome) of the population is a key issue in genetic algorithm. In SDSG, we are using fixed length integer number encoding where feasible solution are encoded in a vector called schedule, of size equal to number of jobs to be scheduled. Each element of the vector is an ordered pair (*Node ID, Security Level*) as shown in Fig. 1. For example *schedule[i]* = (4, 6) means i^{th} *task* is scheduled on *Node ID* 4 and will be executed with *security level* 6.

Task ID	1	2	3	4	5	6
	2, 3	3, 5	4, 6	3, 1	4, 3	2, 8

Fig. 1. The fixed length integer number encoding pattern of chromosomes

C. Initial Population

For initial population, we keep on generating random schedules till we fetch 100 schedules having total security overhead within constrained value denoted as SO_{ctr} . It is obtained by averaging the security overhead from randomly generated 500 schedules and is calculated as shown in equation 4 and 5.

$$SO_{ctr} = \sum_{K=1}^{m} SO_k \Big/ m \qquad (4)$$

$$SO_k = \sum_{i=1}^{n} SO_{ij}, \forall j \qquad (5)$$

where, SO_k is the total security overhead of the k^{th} schedule, *m* is number of schedules in the population which is set to 500 and *n* is the number of tasks comprising the schedule.

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

D. Fitness function and Selection

Fitness function is one of the important components of GA to measure the quality of solution and is problem dependent. The aim of SDSG is to maximize the security of the solution with security overhead as constraint, so fitness of the schedule is the total security being offered to all the tasks and is measured by equation 6.

$$\text{maximize } Fit(f(x)) = \sum_{i=1}^{n} SL_i \qquad (6)$$

$$\text{subject to } SO(x) \le SO_{ctr}, i = 1, 2, ..., n$$

$$\text{where } SO(x) = \sum_{i=1}^{n} SO_{ij}, \forall_j$$

where SL_i is Security level of i^{th} task, $SO(x)$ is security overhead of the schedule x and SO_{ij} is security overhead of i^{th} task on the corresponding j^{th} node. After we compute the fitness $Fit(f(x))$ of each chromosome in the current population, parents for the next generation are selected. The selection operator allows the algorithm to take biased decision favoring good individuals while changing generations. For this, good individuals are replicated while bad individuals are removed. As a consequence, after the selection the population is likely to be dominated by good schedules. We are using roulette wheel selection, which is similar to the roulette in the gambling games. Each individual is assigned an interval proportional to its fitness and is selected if the randomly drawn number belongs to its interval. Better the fitness, better the odd of its being selected. We assume that $P(i)$ is the selection probability of individual i, and is given by

$$P(i) = \frac{Fit(f(i))}{\sum_{i=1}^{N} f(x)} \quad ,i=1,2,..,n \qquad (7)$$

$$partSum(i) = partSum(i-1) + P(i),\ i = 1,2,\ldots,n \qquad (8)$$

where, *partSum* accumualtes partial sum of fitness values. Then we produce a random number R, which is distributed between 0 and 1, if $partSum(i-1) < R < partSum(i)$, we get i as a probable parent. The operation is repeated till we obtain N probable parents for the next generation.

E. Crossover and Mutation

Genetic algorithms are based on principles that crossing two individual can result in offspring's normally better than both the parents. Crossover is a recombination operator that combines subparts of two parent chromosomes to produce offspring that contain some parts of genetic material from both the parents. We have adopted single-point crossover method with probability $p_c=0.7$. Firstly, parents are selected based on the above mentioned selection scheme, then a crossover point is randomly selected, and we exchange chromosome beyond this point. Since the genes of our chromosome are ordered pair of *Node ID* and *Security Level*, crossover not only changes the task node relationship but also the security value associated with it and may result in a new schedule whose security overhead exceeds the SO_{ctr}. Only schedules not exceeding SO_{ctr} are permitted to crossover.

After selection and crossover, mutation is performed to lead the search to get out of local optimum. The mutation operation randomly selects a gene in a chromosome, and then mutates its value. In our case the mutation operator changes the node and security value of a randomly selected task in an arbitrary chromosome with probability $p_m=0.06$. Only schedules not exceeding SO_{ctr} are permitted to mutate.

F. Termination

The entire process of selection, crossover and mutation is repeated for 1000 generations and candidates having best fitness in the final generation is considered as near optimal solution.

VI. Experimental Results and Observations

To validate and evaluate the performance of the proposed SDSG over existing algorithms i.e. MinMin, MaxMin, SPMinMin and SPMaxMin, a simulator in Java is designed and the experiments are carried out. The simulator is composed of Grid-Generator and Task-Generator which randomly generates heterogeneous grid nodes and task sets over the range specified in Table 2. For random value generation, the random function of java API is used which uses current time as the seed value. The experimental study considers the complete heterogeneous environment e.g. security offered by the nodes, speed of the nodes and size of the task. Since MinMin and MaxMin are not security-offering scheduling algorithms; we imposed security overhead cost in their implementations for a fair comparison with SDSG, SPMinMin and SPMaxMin.

The aim of SDSG is to optimize the total security of the schedule, while constraining the security overhead and is made to terminate after 1000 generations. The schedule offering the best security in the final generation is selected as the optimal schedule. For the comparisons to be fair, the other algorithms are compared for the same security value as achieved by SDSG over the following performance metrics:

- *Security Overhead* (equation 2)
- *Makespan* (completion time of the entire job) = $Max\ [CT_{ij,}]\ i=1, 2,\ldots n.$
- *Average response time* (time period between the task arrival and its completion time) $= \sum_{i=1}^{n} (CT_{ij} - AT_{i}) / n$.

where, CT_{ij} is the completion time of the i^{th} task on the assigned j^{th} processor, AT_i is the arrival time of the i^{th} task and n is number of tasks in a schedule. Makespan reflects the entire job efficiency whereas average response time indicates the performance of majority of the tasks within the

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

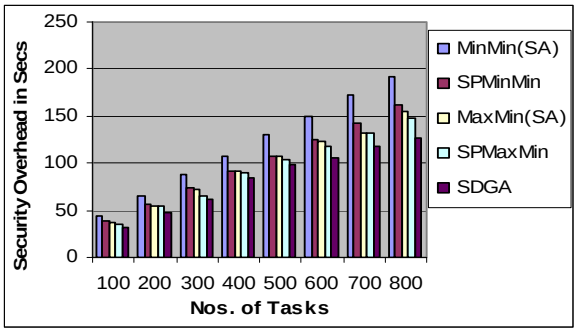
schedule. A better makespan is an indication that the schedule does not suffer and a better average response time suggests that majority of tasks does not suffer.

TABLE 2
INPUT PARAMETERS FOR THE SIMULATION EXPERIMENTS

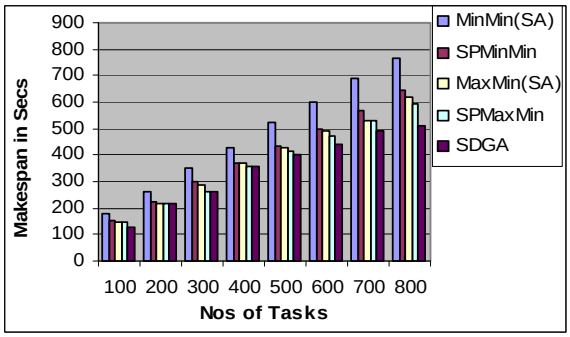
Parameter	Value Range
No of nodes	30
Speed of the processing node (SP)	1, 2, 5, 10 (MIPS)
Security level of the processing node (SL)	1 to 15
No. of tasks	100 to 300 (fixed 150)
Size of tasks	10 to 1000 (MB)
Population size	100
Max generations	1000
Crossover probability(p_c)	0.7
Mutation probability(p_m)	0.06

A. *Performance Impact by varying Number of Tasks*

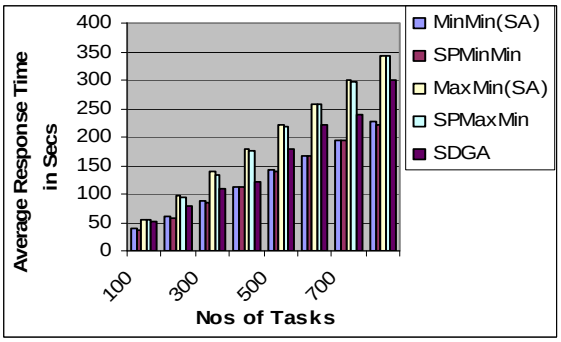
We compared the security overhead, makespan and average response time of SDSG with other heuristics varying number of tasks to be scheduled from 100 to 800, on a grid with 30 heterogeneous nodes. In Fig. 2(b), the experimental results show that when the number of tasks increased, the time for finishing tasks increase too. It is also observed that SDSG performs better than other algorithms, achieving less security overhead and makespan for similar security values. Since MinMin and MaxMin are non security aware schedulers, making them run with higher security value resulted in higher security overhead and makespan. SPMinMin and SPMaxMin are security aware schedulers and give priorities to higher security demanding tasks but does not optimize security overhead. The improvement of SDGA over other algorithms was better with increase in number of tasks as shown in Fig. 2(a). and 2(b). Average response time measures the overall wait time for the entire task set. Since MinMin gives priority to smaller task it shows the best response time and SDGA is the second best among all the algorithms compared for average response time metric.



(a) Security Overhead



(b) Makespan



(c) Average response Time

Fig. 2. Performance comparisons varying the number of tasks

VII. CONCLUSION

In this paper, we proposed a GA based scheduling algorithm for large computational grid, which makes efforts to incorporate security into task scheduling. Non security aware algorithm do not consider security overhead and security constraints of a task and therefore possibly assign the task to a node that only result in small computation time but with a large total execution time. GA, due to its very nature, is capable of searching within the whole range of search space making them global near optimal scheduling solution. The proposed SDSG being security aware genetic algorithm makes effort to optimize quality of security and at the same time satisfy high level of performance metric i.e. security overhead. Experimental results confirm that SDSG performs better than other compared heuristics giving better makespan and security overhead for same level of security.

REFERENCES

[1] Foster I. Kesselman C. Tsudik G. Tuecke S. Security Architecture for Computational Grids. ACM Conference on Computers and Security 1998; 83-91.

[2] Foster I. Kesselman C. Tuecke S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications 2001; pp. 200-222.

[3] Foster I. "What is Grid? A three point checklist,"http://www-fp.mcs.anl. gov/~foster /Articles /WhatIs The Grid. pdf [2004].

[4] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, pp. 671-680,May 1983.

[5] J. H. Holland, *Adaptation in Natural and Artificial Systems.* AnnArbor, MI: Univ. of Michigan Press, 1975.

[6] E. Bonabeau, M. Dorigo and G. Theraulaz, "Inspiration for Optimizationfrom Social Insect Behavior," *Nature*, vol. 406, pp. 39-42,Jul. 2000.

[7] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization,"*Proc. IEEE Int'l Conf. Neural Networks (ICNN 95)*, Perth, Australia,pp. 1942-1948, Nov. 1995.

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

[8] Doulamis, N. Doulamis A. Varvarigos E. Varvarigou T.(2007) 'Fair scheduling algorithms in grids', IEEE Transactions on Parallel and Distributed Systems 18 (11), pp 1630-1648.
[9] Hai, Z., Yuping, W.(2008) 'Security-Driven Task Scheduling Based on Evolutionary Algorithm. International Conference on Computational Intelligence and Security'.
[10] Braun, T., Hensgen, D,. Freund, R., Siegel, H., Beck, N., Boloni, L., Maheswaran, M., Reuther, A., Robertson, J., Theys, M., Yao, B.(2001). 'A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems', Journal of Parallel and Distributed Computing, pp 810-837.
[11] Abawajy, J. An efficient adaptive scheduling policy for high performance Computing, Future Generation Computer Systems 25 (3), 364-370., (2009).
[12] Kalantari, M., Akbari, M. K.(2009) A parallel solution for scheduling of real time applications on grid environments, Future Generation Computer Systems 25 (7), pp704-716.
[13] Kun-Ming, Y., Cheng-Kwan, C.(2008) 'An Adaptive Scheduling Algorithm for Scheduling Tasks in Computational Grid', Seventh International Conference on Grid and Cooperative Computing .
[14] Casanova, H. and Dongara, J.(1996) ' NetSolve: A Network Server for Solving Computational Science Problems'. In Proceedings of the 1996 ACM/IEEE Supercomputing Conference'.
[15] Paranhos, D., Cirne, W., & Brasileiro, F.(2003) 'Trading cycles for information: Using replication to schedule bag-of-tasks applications on computational grids'. In International Conference on Parallel and Distributed Computing (Euro-Par), Lecture Notes in Computer Science, volume 2790, pp169–180.
[16] Fujimoto, N., & Hagihara, K.(2003) 'Near-optimal dynamic task scheduling of independent coarse-grained tasks onto a computational grid'. In 32nd Annual International Conference on Parallel Processing (ICPP-03), pp. 391–398.
[17] Freund, R. F., Gherrity, R. M., Ambrosius, S., Campbell, M. Halderman, D. Hensgen, E. Keith, T. Kidd, M. Kussow, , Lima, J. D., Mirabile, F. L., Moore, L., Rust, B., & Siegel, H. J.(1998) 'Scheduling resources in multi-user, heterogeneous, computing environments with smartnet', In the 7th IEEE Heterogeneous Computing Workshop (HCW'98),pp. 184–199.
[18] Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L., Tuecke, S.(2003) 'Security for Grid Services, Proc. Int'l Symp. High Performance Distributed Computing (HPDC-12).
[19] Xie, T., Qin, X. (2005) "Enahancing Security of Real-Time Applications on Grids through Dynamic Scheduling", Proc. 11th Workshop Job Scheduling Strategies for Parallel Processing JSSPP; pp146-158.
[20] Xie, T., Qin, X.(2008) 'Security-Aware Resource Allocation for RealTime Parallel jobs on Homogeneous and Heterogeneous Clusters'. In IEEE Transactions on Parallel and Distributed systems, Vol. 19, No. 5.
[21] Song, S., Kwok, Y., K. & Hwang, K.(2005) "Trusted Job Scheduling in Open Computational Grids: Security-Driven Heuristics and A Fast Genetic Algorithms," Proc. Int'l Symp. Parallel and Distributed Processing.
[22] Kashyap, R., Vidyarthi, D. P. (2009) 'Security Prioritized Computational Grid Scheduling Model: An Analysis. International Journal of Grid and High Performance Computing', 1(3), pp. 73-84 (2009).
[23] Kashyap, R., Vidyarthi, D. P.(2009) 'A Security Prioritized Scheduling Model for Computational Grid'. In International conference at HPC Asia. pp 416-424.
[24] Xie, T., Qin, X.(2007) 'Performance Evaluation of a New Scheduling Algorithm for Distributed Systems with Security Heterogeneity'. J. Parallel Distributed. Computing ; pp1067– 1081.
[25] Xiaoyong, T., Kenli, L., Zeng, Z., Bharadwaj, V.(2010) 'A Novel Security-Driven Scheduling Algorithm for Precedence Constrained Tasks in Heterogeneous Distributed Systems', IEEE Transaction on computers Vol. 6, No. 1.
[26] Chao-Chin W. Ren-Yi S.(2010) 'An integrated security-aware scheduling strategy for large-scale computational grids'. Future Generation Computer Systems, pp198-206.
[27] Luo Q. Lin Y. Analysis and Comparison of Several Algorithms in SSL/TLS Handshake Protocol . Proceedings of the International Conference on Information Technology and Computer Science 2009.
[28] Stallings W. Cryptography and Network Security: Principles and Practice, 4/E. Prentice Hall: 2008.
[29] Salter M. Rescorla E. Housley R. RFC 5430 Suit B Profile for Transport layer Security and TLS version 1.2. http://tools.ietf.org/html/rfc5430 [March 2009].
[30] Dierks T. Rescorla E RFC 4346 The Transport layer Security (TLS) Protocol Version 1.1. http://tools.ietf.org/pdf/rfc4346.pdf [April 2006].
[31] Xie, T., Sung, A., Qin, X.(2005) 'Dynamic Task Scheduling with Security Awareness in Real-Time Systems', Proceedings of the 19th International Parallel and Distributed Processing Symposium (IPDPS'05), the 4th Int'l Workshop on Performance Modeling, Evaluation, and Optimization of Parallel and Distributed Systems, IEEE/ACM, Denver, CO.

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)