Data Intensive Dynamic Scheduling Model and Algorithm for Cloud Computing Security

Md. Rafiqul Islam[1,2]
[1]Computer Science Department, American International University Bangladesh, Dhaka, Bangladesh
[2]Computer Science and Engineering Discipline, Khulna University, Khulna, Bangladesh
Email: dmri1978@yahoo.com

Mansura Habiba
Computer Science Department, American International University Bangladesh, Dhaka, Bangladesh
Email: mansura.habiba@gmail.com

Abstract—**As cloud is growing immensely, different types of data are getting more and more dynamic in terms of security. Ensuring high level of security for all data in storages is highly expensive and time consuming. Unsecured services on data are also becoming vulnerable for malicious threats and data leakage. The main reason for this is that, the traditional scheduling algorithms for executing different services on data stored in cloud usually sacrifice security privilege in order to achieve deadline. To provide adequate security without sacrificing cost and deadline for real time data-intensive cloud system, security aware scheduling algorithm has become an effective and important feature. Existing systems also merely provide efficient security aware scheduling and security for data. In order to ensure adequate security for different data storages in cloud, in this paper we have proposed a three tier security framework. We have analyzed mathematically the security overhead for different security services such as confidentiality, integrity as well as authenticity and shown that our framework can provide adequate level of security and enhance the processing speed of security services without taking additional overhead in time. We have also proposed a scheduling algorithm to ensure security for data intensive applications. The simulation results show that the proposed algorithm gives better success rate with adequate security in comparisons with existing algorithms. This algorithm ensures security of data and applications as well as confirms the job to be scheduled within deadline.**

Index Terms—**Data Intensive, Security, Cloud, data Storage, Scheduling Algorithm**

I. Introduction

To provide services through Cloud Computing three models are needed. The first two are Infrastructure as a Service (IaaS) provides hardware, software as well as equipment and Platform as a Service (PaaS) includes OS and middleware. The third one is Software as a Service (SaaS) delivers special software [1]. In economic view point cloud computing has gained widespread acceptance. However, without appropriate security and privacy solutions designed for clouds, cloud computing becomes a huge failure [2, 3]. Amazon provides a centralized cloud computing consisting simple storage services (S3) and elastic compute cloud (EC2). Google App Engine is also an example of cloud computing. While these internet-based online services do provide huge amounts of storage space and customizable computing resources. It is eliminating the responsibility of local machines for storing and maintenance of data. Considering various kinds of data for each user stored in the cloud and the demand of long term continuous assurance of their data safety, security is one of the prime concerns for the adoption of cloud computing. In cloud computing a customer will not know where his/her data will be stored, so it is important that cloud service providers are responsible for the integrity and availability of user's data [1]. There are different security issues in all the service levels of cloud computing. Subashini S. and Kavitha V. pointed out that several important security issues in complete cloud environment, which are related to security of third party resources, application security, data transmission security and data storage security [4]. Security of confidential data (e.g., SSN or Credit Card Number) is very important area of cloud computing concern as it can make way for very big problems if unauthorized users get access to it.

In cloud computing, it is very common to store data of multiple customers at one common location. Cloud computing should have proper techniques where data is segregated properly for data security. Care must be taken to ensure that one customer's data does not affect other's data. Again, although data in cloud is stored in shared memory, its owner should have full control and can access their data from any location. Moreover, to provide adequate security in cloud computing various security services such as confidentiality, authenticity and integrity must be ensured [4]. To ensure these services, cryptographic approaches and usage policies must be considered. Proper security services should be used to protect data from unauthorized access and to maintain integrity of them.

doi:10.4304/jcp.9.8.1796-1808

From the above mentioned view of researchers we can cite that the security of data at rest and in transmission is one of the most important issues in cloud computing. The security goals that are related to data security are confidentiality, authenticity and integrity. The different types of data including high sensitive are stored in cloud storage. Considering the sensitivity, all data do not need same level of security. In reference [5] Xie T and Qin T wrote that security can be gained at the cost of performance degradation. On the other hand, according to reference [6] high level of security is reciprocally proportional to system performance and maintenance cost. Hence, if all data storages have to be provided with the highest level of security, it would degrade the performance of the system. So here we have proposed a framework to provide appropriate level of security to different data according to their class of sensitivity with respect to confidentiality, integrity and authenticity.

The Ant Colony Optimization (ACO) [7, 8] algorithm is inspired by the natural context aware and collaborative behavior of real ants in the wild. Recently this algorithm has been widely adopted for large scale complex scheduling problems and has been proved quite impressive, especially in distributed and dynamic computing environment. Z. Wu *et al.* [8] have recommended ACO based task scheduling algorithm as the best candidate for scheduling workflow problem due to its better performance in regards rate of make span time, cost and CPU time. In this paper we have also proposed a scheduling algorithm based on ACO in order to schedule different applications considering their requirement for different level of security. Traditional, scheduling algorithms emphasize on deadline with compromised performance and security. To overcome these limitations of existing scheduling algorithms, the main purpose of our scheduling algorithm is to schedule different jobs in cloud system without compromising system performance as well as security requirement.

II. Architecture and System Design

We explored the three layered architecture proposed by M R Islam and M Habiba [9] with respect to confidentiality and have a modified the three layered architecture in order to provide security services to data storage. In this security framework, we have introduced four different intelligent ants in different layers. They are UA (User Ant), CSPA (CSP Ant), CDSA (CDS Ant) and TA (TrustAgent). Here we have named data storages as honey pots which are labeled as Cl (classified), C (confidential), S (secret), MS (more secret) and TS (Top secret).To make the cloud system more robust, secure and trustworthy we have designed three tiers system architecture. These tiers are User Domain, CSP Domain and CDS Domains shown in Fig. 1, the bottom layer of the framework consists of CU and respective UA. In the middle layer CSPA resides inside CSP along with TA. On the top player Cloud Data Storage (CDS) along with CDSA are placed.

The proposed framework provides security according to the users' demand. For example, if any data owner asks for higher level of security the system provides that. Based on classification of data different levels of security are provided to the data. In Fig.1, it can be seen that honey pots contain data of all five classes. Moreover, each of the classes associated with its definite security level. In this respect, to determine the trustworthiness of the users the proposed framework architecture has introduced the concept of intelligent agents associated with each of the layers of the architecture. The collaborative and self-healing natural behavior of ants has inspired the proposed security system, so each intelligent agent has been considered as an individual ant for the proposed system.

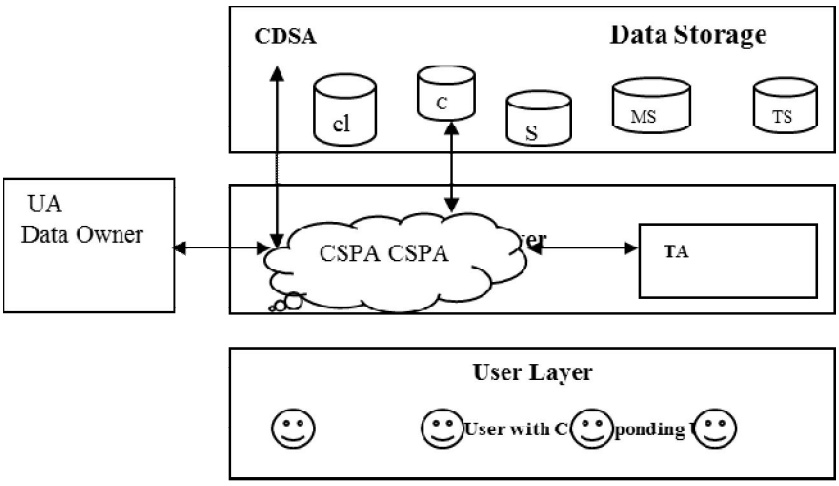

Figure 1: System Architecture for three tier security framework

A. The Activities of Different Ants

In the proposed security framework we have used four different ants with different responsibilities. The activities of different ants are as following.

UA records communication history between user and TA in CSP layer. It stores identical information along with updated trust value. It is also responsible to response to TA's enquiry and participates in the negotiation with agent in Cloud Data Storage (CDS) layer in order to confirm the requirement for security for data storage.

CSPA controls the jobs and migrates them from local to remote data storage. CSPA also takes care of scheduling of jobs. Dispatching job to different cloud droplets is also its responsibility.

CDSA checks trust degree of user domain as well as data owner and updates corresponding trust metrics of corresponding data owner [9]. Sharing trust metrics among neighbor CDSA and TA in CSP layer is also its responsibility. It is in charge of performing negotiation along with the data owner in order to determine the security level for the data storages.

TA is responsible for defining the security level of data storage. Data owner communicates CSPA directly or via any third party to assign an appropriate security level for its owned data storage. CSPA redirects the request for changing or assigning security level of any honey pot in cloud data storage to TA. TA updates the security level for corresponding data storage. Another responsibility of this agent is to impose appropriate authentication method according to defined security level.

All of the aforementioned ants are able to act independent of back-end server as well as to dynamically and continuously assess, configure and remediate security of computing infrastructure.

B. Mechanism for Defining Security Level for Storing Data

Before storing data in cloud data owner should negotiate about the security level for corresponding data with TA resides in CSP layer. TA will define the security level for the data. Defining Trust level comprises of the following steps:

1. Data owner sends requests to CSP layer directly or through any third party.
2. TA checks whether the data owner is valid and authenticated.
3. If the data owner is authenticated, TA redirects the request to data storage layer.
4. Therefore, TA on CSP layer determines the security level for the service, data or application.
5. TA reconfigures the assigned security level for the corresponding data storage and classified the data as one of the five different security levels.
6. TA also updates the level of security of data storage whenever as per request of the owner.

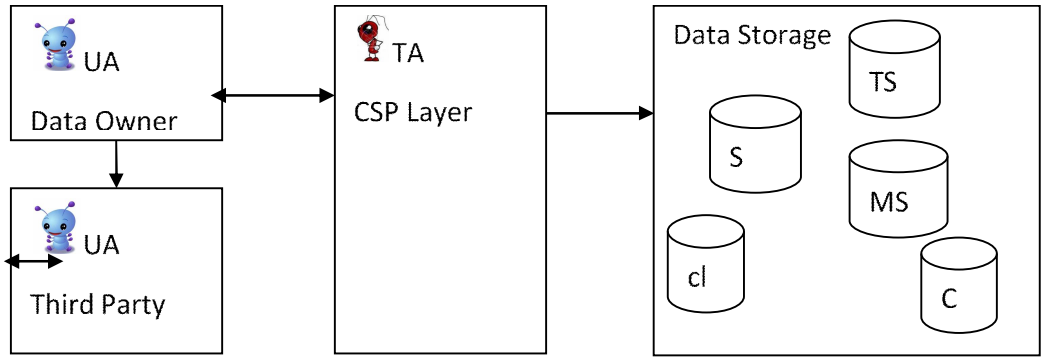

Figure 2: Security Level Determinations for Data Storing

In Fig.2, it can be seen that request for assigning or updating (if necessary) security level of different data storages is coming from either user, e.g. data owner or any third party to CSP layer. TA resides in CSP layer, is responsible for defining security level of data storage. If TA finds that the data owner is authenticated for the requested security level of data storage, TA places corresponding data to the storages that have requested security level.

III. Data Classification and Security Services

To provide different levels of security we have classified data according to their sensitivity levels. Here, data are classified as Top Secret (TS) - the most sensitive one, More secret (MS), Secret (S), Confidential (C) and Classified (Cl). The data of TS class needs highest level of security, on the other hand, the data of Cl class requires lowest level of security. Our target is to ensure adequate level of security for all types of data and optimize the performance of the system. The data are classified according to the policy determined by TA which has been described in section II. For different classes of data different the values of security levels from 0.2 to 0.9 are considered in Table I.

Table I.

Security Levels for Classes of Data

Class of data	Cl	C	S	MS	TS
Required Security level	0.2	0.4	0.6	0.8	0.9

Xie T and Qin X has suggested in their paper [5] that security can be gained at the cost of performance degradation. Hence it is very necessary to determine proper security framework that provides appropriate security level of various security services to applications

and data so that overall performance can be enhanced. That is why; to provide security to data we have considered three security services such as *confidentiality, integrity and authenticity*. These three services are mostly suggested for security of data in cloud. In this regard, we have considered five encryption algorithms such as Rijndael [10], DES [11], Serpent [12], IDEA [12] and 3DES [13] to provide confidentiality to data storages according to the sensitivity levels of data. Similarly, to provide data integrity to five classes of data five different hash functions have been considered such as RIPEMD [14], Tiger [15], Snefru-128 [16], WHIRLPOOL [17], Snefru-256 [16]. Finally, authentication can be provided by message authentication code and five authentication methods have been considered for this purpose. The Message Authentication Code (MAC) methods are UMAC [18], HMAC-MD5 [19], HMAC-SHA-1 [19], CBC-MAC-AES [20] and CCM [20]. Table II shows different algorithms considered for different services with respect to security levels. Here algorithms have been considered according to their security level described in [5, 21] and the performance (speed) cited in [6].

TABLE II.
SECURITY SERVICES AND CHOSEN ALGORITHMS

Service	**Algorithm for different security level**				
	Cl	***C***	***S***	***MS***	***TS***
Confidentiality	Rijndael	DES	Serpent	IDEA	3DES
Integrity	RIPEMD	Tiger	Snefru-128	WHIRLPOOL	Snefru-256
Authenticity	UMAC	HMAC-MD5	HMAC-SHA-1	CBC-MAC-AES	CCM

According to Table I and Table II, each of the algorithms is associated with corresponding security level for data classes Cl to TS. For example, 0.9 is defined to be highest level of security, which corresponds to the cryptographic algorithm 3DES and provides security to the data of Top Secret class. We have considered 3DES to provide highest security among all cryptographic algorithms according to the suggestion of scientific survey on cryptographic algorithms [6, 21, 22]. Conversely, security level of data in Cl is the lowest, which corresponds to the algorithm Rijndael and provides lowest level (0.2) of security. Similarly, among the hash functions Snefru-256 is associated with security level 0.9 to be provided with highest level of integrity, whereas RIPEMD is associated with security level 0.2 and provides lowest level of integrity. Five different authentication methods have been considered for data of in classes Cl, C, S, MS and TS security levels respectively, where weakest authentication method (UMAC) corresponds to data of Cl class and the strongest authentication method (CCM) has been considered for highest security level 0.9 for data in class, TS.

In typical cloud system, if the system selects cryptographic algorithm, which corresponds to the mid level of security such as Serpent. All data are encrypted using the same algorithm. This mid level security might be sufficient for 60% of data, but other 40% data are still vulnerable for malicious attacks. On the other hand, if an algorithm is considered to provide the highest level of security the overall performance of the system may degrade. Therefore, using same cryptographic algorithm for all data storages in cloud is not a suitable solution. As a result, in this paper we have proposed algorithms of different security services for different levels of security for securing data storages.

A. Performance Analyses

Providing security to a system always degrades the performance. We refer performance as data processing speed of the system. Less processing time means good performance. The higher the security requirement is, the lower the performance. Therefore, to provide adequate level of security to the data, labeling data storages in cloud according to different level of sensitivity will enhance the performance of the system. In this section we discuss how data-intensive security framework can enhance the performance (processing speed) and reduce the time taken for different security services in the cloud.

For any cryptosystem, processing speed and time are the most important parameters for its performance [22]. To enhance the performance we have to increase speed without taking time overhead for any security service. For performance enhancement, we have focused on mainly speed and time required for three security services described in previous section. Speed, V for each security service can be formulated as follows:

$$V = \sum_{k=1}^{n} P_k \times V_k \quad (1)$$

Here P_k is the probability of data to be of *k*th class and P_k can be any value within [0.1, 0.9] range, V_k is the speed of service which corresponds to kth level of security. Moreover, time taken for data operation for each security service (T) can be formulated as:

$$T = \sum_{k=1}^{n} P_k \times T_k \quad (2)$$

Here T_k is the time taken by any service. For example,P_1 is the probability of data to be of classified (Cl) type and V_1 is the speed of any service such as confidentiality, integrity and authentication for data of Cl class. Therefore, here V_1 can be the speed of Rijndael algorithm to provide confidentiality service on Cl type of data**.** According to the definition of probability we can write.

$$\sum_{k=1}^{n} P_k = 1 \quad (3)$$

Moreover, the weight for different security classes can be represented as following in ascending order $V_1 > V_2 > \cdots > V_n$

If we provide security according to classes of data, the performance of the system, *V* should be computed as follows:

$$V = P_1V_1 + P_2V_2 + \cdots\cdots\cdots + P_nV_n (4)$$

If we provide highest level of security to all classes of data, the performance of the system should be computed as follows:

$$V' = P_1V_n + P_2V_n + \cdots\cdots\cdots + P_nV_n \quad (5)$$

To get better performance we have to prove

$$V > V' (6)$$

In other hand we can write,

$$P_1V_1 + P_2V_2 + \cdots + P_{n-1}V_{n-1} > P_1V_n + P_2V_n + \cdots + P_{n-1}V_n (7)$$

Since $V_1 > V_n$, $V_2 > V_n$,… …. …$V_{n-1} > V_n$, so

$P_1V_1 > P_1V_n$, $P_2V_2 > P_2V_n$, …….. , $P_{n-1}V_{n-1} > P_{n-1}V_n$

Now we can write

$$P_1V_1 + P_2V_2 + \cdots + P_{n-1}V_{n-1} > P_1V_n + P_2V_n + \cdots + P_{n-1}V_n$$

So,

$$P_1V_1 + P_2V_2 + \cdots + P_{n-1}V_{n-1} + P_nV_n > P_1V_n + P_2V_n + \cdots + P_nV_n$$

So, we proved the inequality "(6)" and "(7)". Here we have considered five classes of data, so $n = 5$ in our case.

Thus we have proved that by providing security according to the class of data we get better performances than providing the highest security to all the data.

B. Security Enhancement for Confidentiality

For discussion we have considered that different sets of data of different classes that occupies different amount (in percentage) space in the storages in cloud as shown in Table III.

TABLE III.

DATA (%) IN CLOUD FOR DIFFERENT SECURITY LEVEL

Class of Data		Cl	C	S	MS	TS
Data (%)	D1	20	10	25	30	15
	D2	10	20	25	25	20
	D3	5	10	40	30	15
	D4	30	30	20	10	10
	D5	10	15	20	30	25
	D6	5	5	15	30	40
	D7	5	5	10	30	50

Confidentiality is one of the most required security services. Here we have found that our proposed data-intensive security framework can enhance the performance of the system significantly. Table IV shows the speed and time taken by different security cryptographic algorithms [23] used for providing confidentiality in our system.

TABLE IV.

PERFORMANCE ANALYSIS FOR POTENTIAL ALGORITHM FOR CONFIDENTIALITY

	Rijndael	DES	Serpent	IDEA	3DES
Speed (MB/sec)	61.01	21.34	21.09	18.96	9.84
Time(sec)	5.378	5.378	5.976	6.499	5.998

Let us consider the amount of data in different classes with respect to row D1 of Table III. Now, from " (1)", the overall speed will be as following.

V = 0.2 × 61.01 + 0.1 × 21.34 + 0.25 × 21.09 + 0.3 × 18.96 + 0.15 ×9.84

= 26.7725 MB/sec

Thus we have computed the speed with respect for rows D2 to D7 of Table III and got speed for respective rows, which are shown in Table V.

TABLE V

COMPUTING SPEED AND TIME USING DIFFERENT DATA SETS

Data in row	Speed (MB/sec)	Time (sec)
D1	26.7725	5.9568
D2	22.3495	5.93175
D3	20.7845	6.0465
D4	31.803	5.6717
D5	21.668	5.9889
D6	16.905	5.7831
D7	16.8345	6.0841
Average	22.4453	5.9232

So, for the different amounts of data in different classes we have found the average speed of cryptographic services as 22.4453. According to the traditional security framework, all data are provided with same level of security. For example, if in security system, the highest level of security is provided and 3DES cryptographic algorithm is considered for providing confidentiality. The speed of system should be the speed of 3DES algorithm, *i.e.*, 9.84 MB/sec. Hence, performance is degraded. Less confidential data are taking unnecessary time. On the hand, in another case, if all data are provided with a medium level of security and the system uses Serpent cryptographic algorithm. The less confidential data will take unnecessary processing overhead. However, the highest confidential data have become vulnerable to malicious attack. Considering all these limitations of traditional security framework, we have proposed data

intensive security framework. In our proposed data intensive security framework, the average speed for confidentiality service is 22.4453 MB/sec. At the same time, the speed enhancement (S.E.) will be as, S.E. = 100 × (22.4453 -9.84) / 9.84 = 128 %

Therefore, implementing data intensive security framework can improve the speed of the system for confidentiality up to 128%. Similarly, the overall time for any particular operation on data can be obtained as "(2)".We can compute time for data set D1 as follows.

T= 0.2 × 5.378+ 0.1 × 5.378+ 0.25 × 5.976+ 0.3 × 6.499+ 0.15 ×5.998 = 5.9568

Therefore, instead of taking 5.998sec for highest confidential level for all data, it requires less time. From Table V we can notice that the average computed time is 5.9232 sec. So, Time reduction (TR) can be measured as,

TR = 100 × (5.9998-5.9232) / 5.9998 = 1.26%.

Hence, we have observed that the speed (performance) of data processing is enhanced as 128% without taking additional time overhead (even with improvement).

C. Security Enhancement for Integrity

Integrity is another important security service. If we consider different algorithms for different levels of integrity, the performance of the system will increase significantly.

TABLE VI.

HASH FUNCTION FOR DIFFERENT LEVEL OF INTEGRITY

Data Class	**Cl**	**C**	**S**	**MS**	**TS**
Hash Function	RIPEMD-320	TIGER	SNEFRU-128	WHIRLPOOL	SNEFRU-256

For simulating the impact of data intensive integrity performance, we have used x86 hash optimization toolkit [23], Intel x86 assembler style and TASM (MASM) x86 assembler syntax to determine the performance evaluation for five different hash algorithms.

Table VII shows the performance evaluation simulation results. All data of the table are measured in different processors, and assume that both code and data reside in the Pentium's on-chip caches (8K each). Under this assumption the figures also scale with the clock speed.

Using the data sets from Table III speeds of processors shown in Table VII and according to the speed of each hash function is assigned for a particular class of data shown in Table VI. For example, SNEFRU-256 is the strongest hash function among five selected hash functions in this paper and it also shows highest time complexity. Therefore, SNEFRU-256 is chosen for highest level of security.

Similarly, we have computed speed for individual data set in case of other processors and the results are shown in Table VIII along with their average.

If we consider highest level of security for all data of row D1 of Table III, the processing speed for iP MMX 23 Mhz, the speed is 2.30 MB/Sec (Table IX), and by considering our framework we have got corresponding average speed, 4.45429 MB/sec.

Hence the speed enhancement = 100 (4.45429 – 2.30)/2.30 = 93.66%. Speed enhancement in case of all the processors has been shown in Table IX. From the Table IX we have observed that the processing speed enhancement is at least 54.40% in case of integrity.

TABLE VII

SPEED FOR DIFFERENT ALGORITHM FOR INTEGRITY USING DIFFERENT PROCESSORS

Processor / Algorithm	**iP MMX 233 Mhz MB/sec**	**K6-2 300 Mhz MB/sec**	**VIA C3 800 Mhz MB/sec**	**iP3 800 Mhz MB/sec**	**iP4 1700 Mhz MB/sec**	**AthlonXP 1330 Mhz MB/sec**
RIPEMD-320	11.28	12.12	24.30	49.77	64.23	87.07
TIGER	8.27	8.25	14.11	29.60	44.38	65.67
SNEFRU-128	3.46	3.59	6.25	11.42	10.21	21.24
WHIRLPOOL	2.35	3.10	4.04	10.75	16.70	18.54
SNEFRU-256	2.30	3.25	4.18	7.64	6.80	14.13

TABLE VIII.

COMPUTING AVERAGE SPEED FOR INTEGRITY USING DIFFERENT DATA SETS AND DIFFERENT PROCESSORS

Data set	iPMMX 233 MHz, Speed (MB/sec)	K62300 MHz, Speed (MB/sec)	VIA C3 800 MHz MB/sec	iP3 800 MHz, Speed (MB/Sec)	iP4 1700 MHz, MB/Sec	Athlon XP 1330 MHz, Speed (MB/sec)
D1	4.9980	5.5564	9.6725	20.1415	25.8715	36.9725
D2	4.6945	5.1845	8.6605	17.9695	23.3915	34.612
D3	3.8250	4.2845	6.965	14.389	17.7715	27.098
D4	7.0220	7.464	13.595	27.935	36.979	53.337
D5	4.3405	4.910	8.0535	16.8385	21.836	31.90
D6	3.1215	3.787	5.742	11.9665	14.695	22.037
D7	3.1785	3.9325	5.8475	12.1605	14.8635	22.388
Average	4.45429	5.01807	8.36229	17.3429	22.211	32.6206

TABLE IX.

SPEED ENHANCEMENT FOR INTEGRITY

Processor	iP MMX 233 Mhz MB/sec	K6-2 300 Mhz MB/sec	VIA C3 800 Mhz MB/sec	iP3 800 Mhz MB/sec	iP4 1700 Mhz MB/sec	AthlonXP 1330 Mhz MB/sec
Average speed	4.45429	5.01807	8.36229	17.3429	22.211	32.6206
Speed for providing highest security	2.30	3.25	4.18	7.64	6.80	14.13
Speed enhancement (%)	93.66	54.40	100.05	127.00	226.63	130.86

TABLE X

PERFORMANCE COMPARISON FOR DIFFERENT ALGORITHMS FOR AUTHENTICITY

Algorithm / Properties	UMAC (Cl)	HMAC-MD5 (C)	HMAC-SHA-1 (S)	CBC-MAC-AES (MS)	CBC-MAC-DES (TS)
Key Size (bits)	128	128	160	128	128
Block Size (bits)	64	512	512	128	128
MAC Time + Key Set Up time (cycle/byte)	3.6296875	37.796875	55.4375	48.734375	90.015625

D. Security Enhancement for Authenticity

Another security service considered here is authenticity. In our proposed security framework, we have found that, for same percentage of different data storages required time span has been reduced up to almost 48% rather than traditional security framework. In Table X, we can find the security enhancement using "(2)".

We have considered the time for PIII –Linux processor only.

Time, $T = 0.2 \times 3.6296875 + 0.1 \times 37.796875 + 0.25 \times 55.4375 + 0.3 \times 48.734375 + 0.15 \times 90.015625$

$= 46.48765625$ cycle/byte

Performance Enhancement, $PE = 100 \times (90.015625 - 46.48765625) / 90.015625 = 48.36$ %

Therefore, the performance enhancement in our proposed system is 128% in case of confidentiality, 54% in integrity and 48% in authenticity based services. The performance enhancement in our proposed system can be visualized in Fig. 3. So, we have shown that the performance will be enhanced in comparison to traditional system.

Figure 3: Performance Enhancements for Different Security Services

IV. Security Constrains Model for Cloud

The main purpose of our scheduling algorithm is to schedule all jobs within their deadline without any performance degradation of the system along with ensuring appropriate security level for each job. In our data-intensive scheduling strategy data will be classified according to the required security level which is already described in earlier section. As the higher security services requires much time and lower security services need lower time, a time balance among different jobs will ultimately increase the system performance in case of the proposed security constraints model. Moreover, determining estimated –execution- time (EET) will be easier and more accurate as the corresponding security service will be defined based on security level. In addition, data owner can change the required security level dynamically after proper negotiation with CSPA. As a result, the relationship between CSP and data owner will be more transparent. This negotiation will ultimately help to reduce the concept of GAP [24] in cloud environment.

High level of security reciprocally is proportional to system performance and maintenance cost [6]. Therefore, in order to impose data intensive real time security as well as to schedule jobs using security aware scheduling algorithm, in this section we propose a security aware job scheduling model to calculate security overhead for different security services. To enhance security with proposed model, we have considered three widely adopted security services for real time system such as confidentiality, integrity and authenticity. Security value for each of the aforementioned security services are defined in the following three sections.

A. Security Value for Confidentiality

The main purpose of confidentiality service is to protect data from outside and unauthorized users. For this data are kept as encrypted, as a result no third party can discover the real data or even cannot understand the embedded algorithms is executable applications. Table II shows that five different encryption methods have been considered for different classes of data or applications in this proposed system. Therefore, the security value of data for confidentiality is a function of amount of data to be encrypted and security level of data. If the amount of data to be encrypted is A_d MB and L_{dc} is the security level of data for confidentiality, which has range 0.2 to 0.9 according to the class of data shown in Table I.

$$SD^c = A_d \times L_{dc} \quad (8)$$

Again the security value of application for confidentiality is a function of amount of algorithm to be made private and security level of application. For A_a amount of algorithm and L_{ac} level of security, the security value of application for confidentiality security service is defined as following.

$$SA^c = A_a \times L_{ac} \quad (9)$$

B. Security Value for Integrity

Integrity security service ensures that no unauthorized user of data or application can temper or modify the application or data. This service is provided through imposing several hash functions. Table VI shows the different hash functions proposed in this paper for different level of security. The security value of integrity can be computed using "(10)", where integrity must be guaranteed for A_d amount of data and L_{dg} is the security level of data for integrity.

$$SD^g = A_d \times L_{dg} \quad (10)$$

Similarly, the security value of application for integrity service can be measured using "(13)", where A_a is the amount of embedded algorithms need to keep save from outside and unauthorized tempering.

$$SA^g = A_a \times L_{dg} \quad (11)$$

C. Security Value for Authenticity

All kinds of accesses to data as well as applications must be authenticated. Table VI shows five different authentication techniques for five different classes of data. As a result, each authentication technique is assigned a security level, L_{da} for data. "(12)" shows the security value of data for authenticity.

$$SD^a = A_d \times L_{da} \quad (12)$$

Moreover, the security value for application in case of authenticity can be defined as following "(13)".

$$SA^a = A_a \times L_{aa} \quad (13)$$

V. Security Value for Overall Security Services

In order to measure the security value of a job in one site we have defined security value for each job to be scheduled using "(14)". Therefore, if i^{th} job is to be scheduled, then the security value SV_i is as following.

$$SV_i = (w^c \times (SA_i^c + SD_i^c) + w^g \times (SA_i^g + SD_i^g) + w^a \times (SA_i^a + SD_i^a)) \quad (14)$$

Here, w^c, w^g and w^a are the priority weights for three security services, such as confidentiality, integrity and authentication respectively. All of the three priority weights can take any value within the range |0 , 1|. If the value of priority weight for any security service is *0*, it means corresponding security service should not be provided for the particular job. For example, if w^c is *0* for any job J_i, it refers that confidentiality security service is not required for J_i job. However, if the value of priority weight for three security service is any value in $0 < w^j \leq 1$, the corresponding security service should be provided to that job, but user can define the priority of different security service. If $w^c > w^a$ for any job j, in case of scheduling the job j, confidentiality will get more prioritize than authenticity. These three weights (values) will be defined by TA in CSP layer after negotiating with UA. If UA asks for any security service such as confidentiality, integrity and authentication, the degree value of corresponding service will be 1 otherwise it will be zero. In addition, Sa_i^c, Sa_i^g and Sa_i^a are the security values for three security services such as confidentiality, integrity and authentication respectively for *i*th job's application. Similarly Sd_i^c, Sd_i^g and Sd_i^a are the security value for the aforementioned three security services for ith job's dataset. These security values are used to assign relative required security level for security services.

A. Security Gain Function

For N jobs the security gain can be computed as "(15)". In conventional distributed computing environment all jobs are provided with either the maximum security level or optimum security level. However in this work, the requested security level has been considered for each job and for the rescheduled job either optimum security level or previously requested security level has been chosen according to the agreement of user. Therefore, the security gain has increased and has a positive impact on the throughput of environment.

$$S^g = \sum_{i=0}^{N} S_i^{opt} - S_i \qquad (15)$$

Here, S_i is the requested security level and S^g is the security gain used for that job with requested security level more than optimized security level.

B. Pricing Issue to Provide Security

Price or cost for to get security services can be defined using the following formula.

$$TP = \sum V(S).C(S).P(S) \qquad (16)$$

Where, $S \in \{c, g, a\}$ and *c* denotes confidentiality, *g* means integrity and *a* denotes authentication.

TP - Total price for the security services.

V(S) – Security value of any service

C(S) – Time units of CPU consumed for execution of a security service.

P(S) – Price of security service of weight 1 (perfect security).

As we have considered the security level from 0.2 to 0.9, since hole-less or perfect security is considered as not possible. We assume that the prices of all security services are not same and we consider the price of integrity service demands 1.2 times of confidentiality service and the price of authenticity service demands 1.5 times of confidentiality. Therefore, if the price for perfect confidentiality service is q, the price for perfect integrity service is 1.2q and the corresponding price for perfect authenticity is 1.5q.

Example: Let's consider there is *b* amount of data and *c* amount of application's algorithms need to be protected for a particular job. The required security level for confidentiality, integrity and authenticity are TS (0.9), S (0.6) and MS (0.8) respectively. As all three services are requested by corresponding job, $w^c = w^g = w^a = 1$. As a result, the security value V(S) can be calculated using Eq. "(6)" to "(10)". Moreover, if the CPU time for execution of services are C_c, C_g and C_a. In addition the price for security services of perfect weight (1) are P_c, P_g and P_a. Then the total cost can be calculated from"(17)".

$$TP = \{0.9 \times (b + c) \times C_c \times P_c\} + \{0.6 \times (b + c) \times C_g \times P_g\} + \{0.8 \times (b + c) \times C_a \times P_a\} \quad (17)$$

VII. Scheduling Algorithm for Real-time Data-intensive Job Scheduling

Xin and Qin has proposed a scheduling algorithm SAREC-EDF [25], where the security level of job is increased and decreased dynamically from the actual requested security level. However, increase in security level decreases the performance and throughput. On the other hand, decrease in security level without user's concern is not acceptable. For this reason cloud service provider (CSP) needs to decrease the security level of any data set or application without negotiating with data owner or user. As a result, users can not have complete faith on CSP, which is a very important problem for cloud computing environment. In this proposed work, this limitation is resolved. This data-intensive security constraint model and scheduling algorithm negotiates with user before changing requested security level.

A. Problem Domain for Scheduling Jobs

At first User submits any request for a security service or application through UA to CSPA, CSPA redirects the request to TA. TA computes required security level for that application or data. Before putting the job into waiting queue TA checks two condition, such as (1) imposing security level will not miss the deadline of the job (2) imposing security level will not impact any subsequent job to be failed. If these two conditions are satisfied, request is put to waiting queue by TA. Otherwise, TA negotiates with appropriate job owner with two proposals such as (1) whether there is any possibility to reduce the required security level or (2) to increase deadline for corresponding job. If job owner agrees any of the two proposals, TA takes appropriate steps to fit requested job within optimum security level. On the other hand, if UA denies both proposals, the job is dropped in to the rejected queue.

B. Scheduling Algorithm

In this paper an ACO based scheduling has been proposed. On the first stage of this scheduling algorithm from line to the best scheduling solution with optimum execution time, cost, make span and requested security level has been produced.

TABLE XI

NOTATION FOR ACO BASED SECURITY SERVICE SCHEDULING ALGORITHM

Notation	Description
EET	Estimated execution time
EST	Estimated starting time
MET	Minimum execution time
EXT	Exact execution time.
D_i	Deadline of *i*th Job
S^g	Security gain
G	mapped Algorithm for corresponding security service in section
S_i^{req}	Requested Security level for ith job
S	Provided security level to job

The ACO algorithm starts with the initialization of all parameters and pheromone. There are three pheromone in this scheduling algorithm, i.e. $\tau_{i,j}$ is the desirability of mapping *i*th task to *j*th resource from the perspective of execution time, $\delta_{i,j}$ is the desirability from the perspective of cost, $\varphi_{i,j}$ is the desirability from the perspective of security. Later the algorithm iterates until the stopping condition set by CSPA is met. According to the basic concept of ACO algorithm [7, 8], at the beginning of each iteration a group of ants are initiated and each ant starts with selecting one of heuristic from time-greedy, cost-greedy, security–greedy or overall-greedy. Then the sequence of task according to the DAG in the waiting queue is built. For each task the requested security services resorted according to the priority weight and corresponding D and EET for the current job as well as the EFT for the predecessors is computed in order to determine whether the schedule can satisfy the precedence relationships defined in the DAG task graph. If the condition is satisfied the job is allocated with suitable resource. Afterwards, the EFT and EST of succeeding jobs are computed. A local update of all pheromone is done to increase the diversity of ACO scheduling algorithm. By contrast if the condition is not met, the job is dropped and put in to the Rejected queue. Later CSPA negotiates with user to fit the job with deadline and security level and put that in waiting queue again. After all ants have built their solution a global update is performed to increase the pheromone towards the so far best solution (*BestSolution*). This global update enhances the convergence of the algorithm. At the end of iteration, *BestSolution* is selected from the solution set according to the user preference. The notations used in proposed algorithm have been described in Table XI and the proposed scheduling algorithm is described in Table XII.

TABLE XII

ACO BASED SECURITY SERVICE SCHEDULING ALGORITHM

1. INITIALIZATION (ACO)
2. While (job scheduling is running)
3. For Each Job in Waiting Queue
4. For each ant TA
5. Calculate the EET, EST, MET, D and SV
6. Sort security services v € $\{c > g > a\}$
7. For each security services, S
8. Select corresponding G
9. SELECTION S_i^{opt} from HEURISTIC
10. If $SV_i^{req} \leq S_i^{opt}$ and EET ≤ Deadline
11. Dispatch the job for scheduling and remove it from Waiting Queue
12. Compute S^g Using "(7)"
13. Compute $S^{overhead}$
14. S=S_i^{req}
15. Until Waiting Queue is Empty
16. Else If EET > D and If (D + $S_i^{req} \leq S$)
17. Reject the task
18. Negotiate S_i^{req} and fit the job to be rescheduled
19. LOCALUPADTE ($S^{overhead}$, speed, time, cost)
20. For each Job in the Dispatcher Queue
21. Get Data from CDS and execute
22. GLOBALUPDATE ($S^{overhead}$, speed, time, cost)
23. Until Dispatcher Queue is Empty
24. Return (Solution, Solution Set)
25. BestSolution =COMPARE (SolutionSet, UserPref)

C. Simulation and Performance Evaluation

We have developed a cloudsim [26] based tool to perform the simulation. The algorithm discussed in previous section has been discussed. The key system parameters of the simulated system are given in Table XIII. A traditional scheduling problem can be defined as a composite of three parameters such as number of tasks (T), number of VM (V) and number of data set (M). However, in data-intensive computing environment there are another two parameters which are a vector of all security levels (L) and a vector containing the percentage

of tasks for each security level (D). Seven different dataset has been selected for simulation. For each dataset all parameter are same except D, in different dataset

TABLE XIII.

SYSTEM PARAMETER FOR SIMULATION

Parameter	Value
Deadline Base Laxity	(100-1000)s
Job Security Level, L	<Cl, C, S, MS, TS>
Security Services	c , a, g
Number of Job, T	1000
Number of VM, V	8-512
Number of data set, M	4-256
Percentage of tasks for each security level, D	D1, D2, D3, D4, D5, D6 and D7 described in table 1

different percentage of job for different security level have been selected.

On the other hand, the performance metrics included success rate, speed enhancement which is measured in CPU time unit, security value and throughput. To demonstrate the performance strength, we have considered three other well-known algorithms used for scheduling in cloud environment such as SARDIG, SAREC_EDF, VNPSO and MPSO.

we want to impose highest security level in traditional system for other three algorithms, the overall speed is 9.84 MB/sec. However, if we implement ACO based algorithm in our proposed system framework, the speed will be significantly increased.

Table XV shows the simulation result for the security value for the algorithms for different percentage of data of different security level for virtual machines ranges from 8 to 512 and data set ranges from 4 to 256. As the proposed algorithm consider security value for both dataset and application code separately and weight them with different values, therefore, it can be seen that proposed DACO algorithm significantly outperforms other algorithms in terms of security value.

Another observation from Table XVI shows that the proposed algorithm also has throughput for four reference algorithms and the proposed algorithms. The success rate depends on the percentage of job those have been scheduled with the requested security level. That means the requested security level is not changed. In case of SARDIG if security level is not met the job is rejected, therefore when the percentage of job with higher security level increases, the success rate of SARDIG is very low. On the other hand, in case of SARC_EDF, if the requested security level is lower than the maximum security level, therefore, if the percentage of lower security level is high, the security level is increased, therefore, for the data set such as D4 and D1the success rate drops significantly. SARC_EDF also does not show better performance with dataset with larger portion of higher security such as D7 and D6. From Table XIV it

TABLE XIV.

PERFORMANCE COMPARISON IN TERMS OF SPEED ENHANCEMENT

D	V	M	Speed Enhancement				
			SARDIG (%)	***SAREC_EDF(%)***	***VNPSO (%)***	***MPSO (%)***	***DACO (%)***
D1	8	4	9.839	9.83925	9.84	9.84	61.01
D2	16	8	9.8311	9.814	9.84	9.84	41.34
D3	32	16	9.8387	9.804	9.84	9.84	21.31
D3	64	32	9.841	9.81	9.84	9.84	21.09
D5	128	64	9.8378	9.81	9.84	9.84	19.84
D6	256	128	9.8152	9.81	9.84	9.84	18.96
D7	512	256	9.7802	9.81	9.84	9.84	9.84

Table XIV shows the speed enhancement for data – intensive security framework. In other three algorithms same security level is chosen for all data. Therefore, selecting lower or medium level of security is very risky; on the other hand, highest security level will decrease the overall performance. For example, for confidentiality, if

can be shown that the success rate of proposed DACO significantly outperforms all other algorithms for all cases. From simulation result, it can be seen that if the higher level of security is requested by most of the jobs those are to be scheduled the proposed algorithm is the better choice among all five algorithms.

TABLE XV.

PERFORMANCE COMPARISON IN TERMS OF SECURITY VALUE

D	V	M	Security Value				
			SARDIG	*SAREC-EDF*	*VNPSO*	*MPSO*	*DACO*
D1	8	4	0.87	0.81	0.88	0.85	0.94
D2	16	8	0.89	0.83	0.90	0.87	0.95
D3	32	16	0.91	0.84	0.92	0.89	0.96
D3	64	32	0.92	0.85	0.94	0.95	0.99
D5	128	64	0.93	0.87	0.96	0.97	1.01
D6	256	128	0.94	0.89	0.98	0.99	1.03
D7	512	256	0.95	0.90	1.00	1.01	1.04

TABLE XVI.

PERFORMANCE COMPARISON IN TERMS OF SECURITY VALUE

D	V	M	Security Value				
			SARDIG	*SAREC-EDF*	*VNPSO*	*MPSO*	*DACO*
D1	8	4	0.87	0.81	0.88	0.85	0.94
D2	16	8	0.89	0.83	0.90	0.87	0.95
D3	32	16	0.91	0.84	0.92	0.89	0.96
D3	64	32	0.92	0.85	0.94	0.95	0.99
D5	128	64	0.93	0.87	0.96	0.97	1.01
D6	256	128	0.94	0.89	0.98	0.99	1.03
D7	512	256	0.95	0.90	1.00	1.01	1.04

TABLE XVII.

PERFORMANCE COMPARISON IN TERMS OF THROUGHPUT

D	DeadLine Base (ms)	Throughput				
		SARDIG	*SAREC-EDF*	*VNPSO*	*MPSO*	*DACO*
D1	0.1	0.87	0.81	0.88	0.85	0.94
D2	0.3	0.89	0.83	0.90	0.87	0.95
D3	0.4	0.91	0.84	0.92	0.89	0.96
D3	0.5	0.92	0.85	0.94	0.95	0.99
D5	0.6	0.93	0.87	0.96	0.97	1.01
D6	0.8	0.94	0.89	0.98	0.99	1.03
D7	1.0	0.95	0.90	1.00	1.01	1.04

VI. CONCLUSION

In this paper, three tiers system architecture is proposed that supports real time data-intensive security services. Three main security services are considered and ordered them according to their security values. The main focus of this security constraint model is to classify data according to required security level. So the system needs different time and performance threshold to provide different security services. In this system, data of lower security level will not be provided with higher security services unnecessarily as like traditional cloud security system. Similarly here the data of higher security level has no risk of losing security threshold due to using of lower level of security services. In this proposed work, performance has been enhanced to significant amount due to classifying data into different categories based on the required security levels. The main goal of this proposed security model is to enhance security as well as performance. Moreover, an ACO based scheduling algorithm is introduced in this work for scheduling jobs for increasing the security level of the environment. The simulation results show that the proposed algorithm

provides better success rate within required security rather than existing algorithms. This algorithm ensures security of data as well as confirms the job to be scheduled within deadline.

REFERENCES

[1] I. Foster, Y. Zhao, I. Raicu, S. Lu, Cloud Computing and Grid Computing 360-Degree Compared, Grid Computing Environments Workshop, pp. 1-10,12-16 Nov, 2008

[2] Y. Chen, V. Paxson, R. H. Katz, What's New about Cloud Computing Security? , Technical Report No. UCB/EECS-2010-5.

[3] H. Takabi and J. B. D. Joshi, security and Privacy Challenges in Cloud Computing Environments, IEEE Security and Privacy, Vol. 8, Issue 6, pp. 24-31, Nov-Dec, 2010

[4] S. Subashini, V. Kavitha, A survey on security issues in service delivery models of cloud computing, Journal of Network and Computer Applications, Vol. 34, Issue 1, pp. 1-11,Jan 2011

[5] Xie, Tao, and Xiao Qin. Enhancing security of real-time applications on grids through dynamic scheduling. In Job Scheduling Strategies for Parallel Processing, pp. 219-237. Springer Berlin Heidelberg, 2005.

[6] The Consultative Committee for Space Data Systems, Encryption Algorithm Trade survey, Information report, CCSDS 350.2-G-1, Green Book, March, 2008.

[7] M. Dorigo, M. Birattari, T. Stutzle,, Ant Colony Optimization-Artificial Ants as a Computational Intelligence Technique, Computational Intelligence Magazine, IEEE, Vol.1, Issue. .4, pp.28-39, Nov. 2006

[8] Z. Wu, X. Liu, Z. Ni, D. Yuan, and Y. Yang. A market-oriented hierarchical scheduling strategy in cloud workflow systems. Journal of Supercomputing, Vol. 63, Issue 1 , pp 256-293, Jan. 2013.

[9] M. R. Islam and M. Habiba, Collaborative Swarm Intelligence based Trusted Computing, in Proc. of International Conference on Informatics, Electronics and Vision (ICIEV'12), pp. 1-6, May 18-19, 2012, Dhaka, Bangladesh.

[10] T. Jamil., "The Rijndael algorithm," Potentials, IEEE , vol.23, no.2, pp.36-38, April-May 2004

[11] T. Nie, and T. Zhang. "A study of DES and Blowfish encryption algorithm." In TENCON 2009-2009 IEEE Region 10 Conference, pp. 1-4. IEEE, 2009.

[12] E. Biham, R. Anderson, and L. Knudsen. "Serpent: A new block cipher proposal." In Fast Software Encryption, pp. 222-238. Springer Berlin Heidelberg, 1998.

[13] G. Na-na, Z. Li, and Q. Wang. "A Reconfigurable Architecture for High-Speed Implementations of DES, 3DES and AES." Acta Electronica Sinica Vol. 34, Issue .8, 2006.

[14] H. Dobbertin, A. Bosselaers and B. Preneel. RIPEMD-160: A strengthened version of RIPEMD. Fast Software Encryption, Springer Berlin Heidelberg, pp. 71-82, Jan. 1996.

[15] F. Mendel, B. Preneel, V. Rijmen, H. Yoshida, and D. Watanabe. Update on tiger. In Progress in Cryptology-INDOCRYPT 2006, pp. 63-79. Springer Berlin Heidelberg, 2006.

[16] M. C. Ralph, A fast software one-way hash function. Journal of Cryptology 3, Vol. 3, Issue. 1, pp.43-58, Jan. 1990

[17] P. S. L. M Barreto and V. Rijmen. The Whirlpool hashing function .First open NESSIE Workshop, Leuven, Belgium. Vol. 13. 2000.

[18] B.V. Rompay, The First Report on UMAC, Belgium, March 2001

[19] M. Bellare,, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication, Advances in Cryptology—CRYPTO'96. Springer Berlin Heidelberg, pp 1-15, 1996.

[20] M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. Journal of Computer and System Sciences, Vol. 61, Issue. 3, pp.362-399, 2000.

[21] M. R Islam, M. T. Hasan and G. M. Ashaduzzaman, An Architecture and A Dynamic Scheduling Algorithm of Grid for Providing Security for Real-Time Data-Intensive Applications, International Journalof Network Management,Issue. 21, pp.402-413, 2011.

[22] Md. Rafiqul Islam and M. Habuiba, Agent Based Framework for providing Security to data storage in Cloud, In proceedings of ICCIT 2012, Chittagong , Bangladesh, pp.446,451, 22-24 Dec. 2012.

[23] T. Xie, X. Qin, Security-driven scheduling for data-intensive applications on grids, Cluster Computing, Vol. 10, Issue. 2, pp.145–153, 2007.

[24] Md. T. Khorshed, A. B. M. Ali, and S. A. Wasimi. A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing. Future Generation Computer Systems, Vol. 28, Issue.6, pp. 833-851, 2012.

[25] Cloudsim toolkit, www.cloudbus.org/cloudsim/, Last Visited: Jan. 2013.

[26] T. Xie, X. Qin, Security-driven scheduling for data-intensive applications on grids, Cluster Computing, Vol. 10, Issue 2, pp.145-153, 2007.

Md. Rafiqul Islam obtained combined BS and MS securing first class with honors in Computer Engineering From Azerbijan Polytechnic Institute (Azerbaijan Technical University) in 1987 and Ph.D. in Computer Science from Technological University of Malaysia (UTM) in 1999. He did Post Doctoral research in Japan Advanced Institute of Science and Technology (JAIST) as a JSPS fellow in 2001. He worked as a Professor, Head of Computer Science and Engineering Discipline and Dean of Science, Engineering & Technology School of Khulna University, Bangladesh. Currently he is working as a Professor in Computer Science Department of American International University-Bangladesh (AIUB). He has published more than 80 papers in National and International Journals as well as in referred International Conference Proceedings. His research interest includes Design and Analysis of Algorithm in the area of Information Security, Image Processing, Grid Computing, Cloud Computing, Bio-informatics, Information Retrieval etc.

Mansura Habiba obtained MSc in Software Engineering from American International University, Bangladesh in 2013 and B.Sc in Computer Science and Engineering in 2007 from Khulna University, Bangladesh. She worked as a Software Engineer, in different companies. Her research interest includes Design and Analysis of Algorithm in the area of Cloud Computing and Utility Computing etc. At present she is trying to get admission as a Ph.D. student in any prestigious university.