SCM: Secured Cloud Model defeating Un-authorization using digital signature with chaotic secure hashing and Work scheduling based on improved Genetic algorithm

[1]Senthil Kumar, [2]Dr.Thabasu Kannan
[1]Research Scholar, [2] Research Supervisor
[1]Department of Computer Science and Engineering,
[1]MAHER University, Chennai, Tamilnadu, India

Abstract : Cloud data allocation facility, which allows a group of user to work together to access and the shared data, is one of the most standard and effective working styles in the enterprises. So, in spite of having advantages of scalability and flexibility, cloud storage service comes with confidential and the security concerns. A direct method to defend the user's data is to encrypt the data stored at the cloud. In this research work, we suggest a Secure Cloud Model (SCM) that contains user authentication, and data scheduling. An innovative digital signature with chaotic secure hashing (DS-CS) is used for user authentication, followed by an enhanced work scheduling based on improved genetic algorithm to reduce the execution cost. We executed the proposed SCM in the Java simulator and estimated its performance in terms of throughput, Normalized schedule cost, Success rate, end to end delay and Number of Packet Loss**.**

IndexTerms **– Cloud Security, Digital signature, Chaotic secure hashing, authentication, work schedule.**

I. INTRODUCTION

Cloud computing is an emerging technology for data transmission mode [1]. It contains huge number of computer through a communication network like Internet for data sharing and storage [2].Cloud computing has the benefit of delivering a flexible, high-performance, pay-as-you-go, on-demand service [3]. Traffic management plays a significant role in minimizing congestion; the OpenFlow protocol is aimed at minimizing packet loss and maximizing the quality of experience (QoE), quality of service (QoS), using resources optimally and estimating end-to-end network performance [8-12]. SDNs are designed to support thousands of applications through manageable network flows and integrated security mechanisms [13-15]. An SDN-based cloud computing environment is open to to distributed denial of service (DDoS) attacks. In addition, because several harmful threats may be introduced into the network, SDNs include software-based traffic analysis, and new flows are analyzed [16, 17]. SDN-based cloud architectures are also vulnerable to security threats, including (but not limited to) unauthorized access, data leakage, data modification, forged traffic flows, vulnerabilities in switches/controllers and malicious applications. Threats can be instigated by different individual attackers whose aim is to degrade network performance [18].

Operators should assurance to the users and stick to the Service Level Agreement (SLA). It will absolutely lead to unproductive if the job spanning is too long. Also, the cloud computing platform also wants to vigorously balance the load among the cloud servers so as to neglect hotspot and develop resource utility [4]. So, how to dynamically and professionally schedule tasks and meet users becomes a critical issue to be solved. The workflow schedule in cloud computing is to realize the mapping between each interrelated workflow task and the available cloud resources [5]. The scheduling objective is to encounter the need of the objective function defined by users. However, in essence, the problem of workflow schedule is a NP problem that can't be solved in polynomial time. Workflow schedule algorithm involves two types: the best-effort scheduling and QoS constraint scheduling [6]. The best-effort scheduling mainly aims for minimizing the time of workflow scheduling, while not considering the cost of resource access. This kind of algorithm mainly is applied to grid computing. Its representative algorithms include minmin, max-min, and suffrage and so on [7].

The remaining section of this paper is structured as follows. Section II reviews earlier studies on cloud security and work scheduling algorithm. Section III illustrates the proposed SCM-cloud environment, with its novel algorithms. Section IV describes the results attained from implementation simulation. Finally, Section V concludes the paper.

II. BACK GROUND STUDY

The QoS constraint scheduling's related researches normally optimize the workflow scheduling under satisfying user-defined deadline or budget constraints. For e.g., the critical path algorithm IC-PCP [19], the enhancement IC-PCP algorithm EIPR [20], the partition uniform time algorithm PBTS [21] and the improved heterogeneous earliest finishing time algorithm BHEFT [22]. The limitation of the above algorithms is that they ignore the dynamic elastic characteristics of the cloud assets provision in addition to the characteristics of resource performance and cost heterogeneity. These workflow scheduling optimization algorithms cannot be

applied directly in the cloud computing environment. Therefore, it is necessary to find a cloud workflow scheduling algorithm which is applicable to cloud computing environment combining task characteristics and resource characteristics.

Typically, a job contains the series of tasks. Task scheduling becomes a huge challenge for efficient scheduling algorithm design and implementation, by means of typical NP-complete problem [23]. All tasks in this paper are computational ones, and independent with each other. There are two solutions for these problems. One is exact solution, for example, cupidity algorithm, branch and bound method, Dynamic Program. Although these algorithms can find a solution exactly, when search space enlarges, it turns out to be poor efficiency. We can adopt special technical to speed it up, but it does not improve the time complexity actually and still cannot meet large-scale problem conditions. Another method is called heuristic algorithm, instead of searching the entire domain, these methods have good efficiency. As a typical, genetic algorithm shows powerful search ability. Researchers have never stopped studying on it [24][25][26][27]. Some researchers propose varying probabilities of crossover and mutation in genetic algorithm [28]. WANG Rong-Fang presents a self-adaptive dynamic control strategy of population size [29]. In cloud computing, CSGA[30] through genetic algorithm schedules task to the node with data block of this task in order to reduce data translation cost, which aims to shorten all the task completion time and tries to improve the consumer satisfaction. Through DFGA [31], the better task scheduling not only shortens total completion time but also shorten average completion time. Without affecting long jobs, a system known as piranha is proposed by Khaled elmeleegy [32] to optimize the short jobs on hadoop. By considering the superior nature of cloud platform, the inter nodes load should be considered. Some research according to historic data and present state of the system and using genetic algorithm proposes a tactic computing gaining the influence it will have on the system after positioning of the needed VM assets and then to process the current task the least-affective VM choosing is occurs. Using this this method balance the loads effectively and prevents the dynamic migration. But still this strategy does not consider job spanning.

The major offerings of this work are shortened as follows.

An integrated, SCM-cloud (cloud based) architecture is designed to resist unauthorized access and work scheduling.

- User authentication is provided using a digital signature with a chaotic secure hashing (DS-CSH) algorithm that authenticates users and mitigates the problem of an unauthorized user entering the cloud environment.
- An enhanced Improved Genetic algorithm (IGA) protocol is applied to improve the quality of service (QoS) of the communications between user and cloud.

III. PROPOSED SYSTEM MODEL

The proposed SCM-cloud targets to solve the problems related with unauthorized access and budget constraint in a cloud framework. The novel SCM-cloud design includes two stage of phase, in first phase, user authentication is evaluated by DS-CS algorithm and second phase is task scheduling using job spanning time and load balancing improved genetic algorithm (IGA). Figure. 1 shows the proposed SCM cloud model.

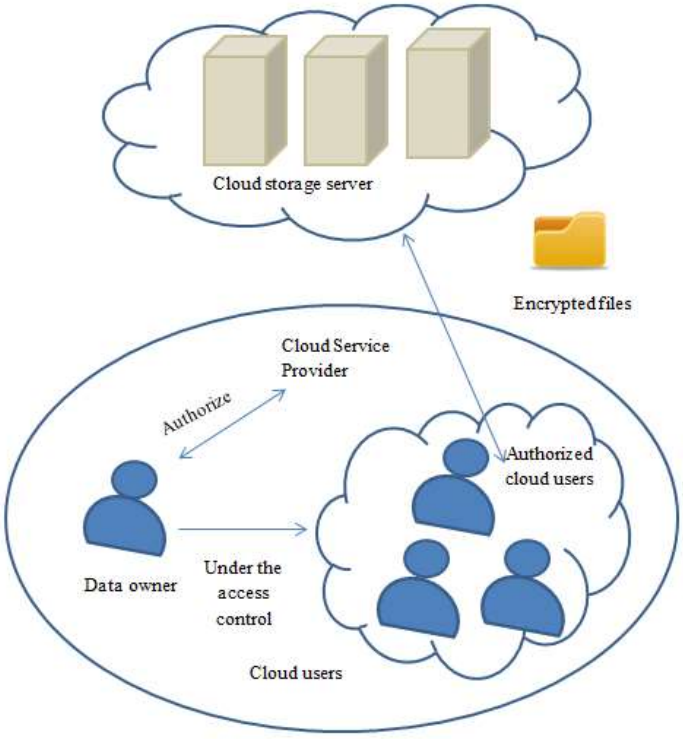

Fig 1. Proposed SCM model

3.1 User Authentication Phase

The user authentication phase authenticates all users looking to file storage in the cloud. The SCM-cloud uses the innovative DS-CSH algorithm to authenticate users based on the originality of their signatures. The secure hashing algorithm (SHA) is broadly

used to produce hash functions for security purposes. The SHA attains integrity, authentication and a digital signature. This research uses SHA3-384, in which "384" denotes the bit length of the digest.

An overview of CBHF module is depicted in Figure 2. In which message block length is M_{n-1} is converted into fixed message digest of H_n with respect to key K_{n-1}.

Chaotic hash functions use mapping schemes to estimate the trajectory and choose valid initial conditions. Chaotic behavior is well-defined by Lyapunov exponents. When a model is recognized as chaotic, the trajectory difference with respect to the initial condition increases with respect to the time period. The difference is explained by following expression

$$d_t = d_0 2^{\lambda t} \tag{3.1}$$

Here, d_o represents the initial condition and d_t represents the distance reached at t and λ defines the Lyapunov exponents, which are expressed by averaging points as below:

$$\lambda = \frac{1}{t^N - t_0} \sum_{k=1}^{N} \log_2 \frac{d(t_k)}{d(t_{k-1})} \tag{3.2}$$

Here N denotes the number of points, t_0 and t_N are the initial time and time at N points respectively. Similarly $d(t_k)$ and $d(t_{k-1})$ are represents the distance at point k and point $k-1$, respectively.

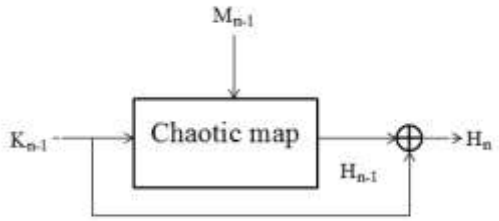

Figure 2 Basic diagram of chaotic hash function

After finding the prediction of $\boldsymbol{\lambda}$, the system is measured either chaotic or not chaotic. When >0, then the system is known as chaotic. The SHA3 algorithm utilizes sponge construction in which the message blocks are undergone by XOR operation for conversion into the permutation function f. Let consider that the input string (IS)is padded and converted into f, the algorithm performs over bit blocks of width b, rate r and output length d. The constructed sponge is illustrated as follows

$$Z = Sponge[f, pad, r](N, d) \tag{3.3}$$

From the sponge bit string, we attain a hash function output of length 384. The use of SHA3-384 does not control the message size. This hashed function is sent to cloud service provider for authentication. When a user is valid, their packets are moved to cloud based on defined policies. Using hash functions, attackers can be identified before their packets enter the cloud.

3.2 Work Scheduling Phase

Genetic algorithms are a class of optimum search algorithms which simulate biological progress and genetic mechanism. After the initial population is created, it develops better and better estimated solutions based on fitness from generation to generation. During every generation, the individual is selected based on the fitness of the different individuals in a certain problem field. Then the individuals combined, cross and vary by the genetic operators in natural genetics and then a new population demonstrating a new solution set is formed. The best solution will be voted, and the poor solutions will be removed after several generations of evolution. Based on the real state of cloud computing, this paper suggests a scheduling strategy through improved genetic algorithm. The basic steps of IGA are shown in Algorithm 1.

Algorithm 1: IGA

Input:

MaxIter: max iterations
S: Population scale; N: Worker node number;
J: Number of jobs
C_1, C2: Weights of total jobs panning and average job spanning, $C_1+C_2=1$
λ_1, λ_2 : Probability of *fitnees*$_1$ and *fitness*$_2$ $\lambda_1+\lambda_2=1$

Output:

elite$_{1,N}$: Best solution

1: *iterator*, $\lambda \leftarrow 0$; *fitness* $\leftarrow \Theta$

2: *elite*$_{1,N}$, *temp*$_{1,N}$, $p_1, p_2, p_{s,N}$, *Fitness* $\leftarrow \Theta$

3: p ← *Greedy Initialization*
4: **While** *iterator* < *MaxIter* **do**
5: λ = random(0,1)
6: **if** $\lambda < \lambda_2$ **then**
7: *fitness* = $Fitness_2$
8: **else**
9: *fitness* = $Fitness_1$
10: **end if**
11: **for** i=1 to S **do**
12: $Fitness_i$ ← fitness(i)
13: **end for**
14: $elite_{1,N}$ ← *individual with best fitness*
15: $p_1, p_2 \leftarrow 0$
16: *calculate* p_c, p_m
17: **if** *random*(0,1) < p_c **then**
18: $temp_{1,N}$ ← *crossover*(p_1, p_2)
19: **end if**
20: **if** *random*(0,1) < p_m **then**
21: $P_{1,N}$ ← *mutation*(*temp*)
22: **end if**
23: *iterator* ← *iterator* + 1
24: **end while**

3.2.1 Representation of Chromosome

Giving to the problematic model termed above, a scheduling schema can be stated by a chromosome that coded in form of cloud user [19]. Length of chromosome (number of genes) is accurately equal to task number. Value of a gene is equal to the node number, on which the task this gene resembles to is implemented. For example, we have 4 worker nodes, 4 jobs, and the jobs are split into 2, 3, 4, 5 tasks distinctly. So there are total 14 tasks. We describe the length of chromosome is 14, task No. varies from 1 to 14, gene varies from 1 to 4. Given the following chromosome

{1,3,2,4,3,1,4,2,3,1,3,2,4,1}

We achieve the first task implements on the first node; the second task performs on the third node. As above stated, the chromosome will be interpreted into following array

$Node_1$: {1,6,10,14}, $Node_2$: {3,8,12}

$Node_3$: {2,5,9,11}, $Node_4$: {4,7,13}

3.2.2 Initialization of population

Basic GAs creates initial population randomly, and individuals have low performance. In this paper, greedy algorithm is used. We create initial population through directing optima choice based on current condition. Now we assume the job t be processed is enough. And there is no priority. In Algorithm 2, w present the greedy initialization algorithm. To remove data transfer to great amount, data locality is taken into account.

Algorithm 2 : Greedy initialization (GI)

Input:
S: Population scale; N: Worker node number;
J: No. of jobs

Output:
$P_{s,N}$ **:** Initialized population

1: $P_{s,N} \leftarrow \Theta$

2: $i, j \leftarrow 1$
3: **While** $i \leq S$ **do**
4: **While** $j \leq N$ **do**
5: **if** There is a node g with k needed data replication on it and g is
6: idle then
7: $site \leftarrow g$
8: **else**
Choose the least cost node as executing node f
9: **end if**
10: $P_{i,k} \leftarrow site$
11: $j = i + 1$
12: **end while**
13: $i = j + 1$
14: **end while**
15: **return** $P_{s,N}$

3.2.3 Fitness Function

In commonly, fitness is the ability to create descendants. Genetic algorithm impersonate survival-of-the-fittest principle of nature to make search procedure[21], during which fitness function is the standard for the quality of the individuals in the population. Operators of selecting, crossover and mutation are all respect to fitness. Fitness function is the mainspring of genetic algorithm.

In Cloud framework task scheduling is a multi-objective optimization problem. In this research paper, we describe two fitness functions. The separate job with less inter-nodes load variance and time consuming has better fitness. First we present weight function to combine jobs' total time spanning and average time consuming. "C1TotalTime(i)+C2AvgTime(i)" indicates the time performance of individual i. To ensure the constant, we take the reciprocal of completion time and load variance respectively. Also, we adopt constant D1 and D2 to prevent individual fitness approaching zero when system load variance coefficient is too huge or long time with no service response.

$$Fitness_1 = \frac{D_1}{C_1 TotaTime(i) + C_2 AvgTime(i)} \tag{3.4}$$

$$Where\ 1 \leq i \leq SCALE, 0 < C_1, C_2 < 1, C_1 + C_2 = 1$$

$$Fitness_1 = \frac{D_2}{\alpha(TotalTime)} \tag{3.5}$$

$$Where\ \alpha(TotalTime) = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left(P(j,TotalTime) - \overline{P(TotalTime)}\right)^2} \tag{3.6}$$

Table 1: Parameter Setting of IGA

Parameter	Value
k_1	0.38
k_2	0.79
k_3	0.099
k_4	0.06
c_1	0.6
c_2	0.4
λ_1	0.7
λ_2	0.3

3.2.4 Strategy selection

Although the rotating selection algorithm, crossover operation and mutation operation ensure the population diversity, the individual with better performance may be destroyed by these operations. To prevent this situation from happening, elitist individual are copied to the next generation without other operators. In this paper, elitism is used to allow some of the better organisms from the current generation to carry over to the next, unaltered. Theories have proved that genetic algorithm with elite selection operator have good global convergence.

In genetic algorithm, it's similar that individuals with good fitness have greater probabilities to be selected. In this paper, double fitness function is accepted. We should decide to take which one as criteria before select a Chromosome. Here we import λ_1 and λ_2 to represent the probability of p_i and p_i' respectively, where $0< \lambda_1, \lambda_2 <1$, $\lambda_1 + \lambda_2=1$. We randomly select p_i (Formula 12) or p_i' (Formula13). Here, λ_1 is greater than λ_2, as we give priority to job completion time than inter-nodes balance. Selection strategy is the guiding factor for genetic performance. The algorithm in this paper uses the selection strategy based on fitness ratio. First of all, we work out the fitness of the individuals in current population according to fitness function, and latter conduct the total fitness summation. Then, for individual i at time T we calculate the selection probability $p_i(t)$ and $p_i'(t)$ among the population according to their fitness values

$$P_i(T)=\frac{fitness_{1i}(T)}{\sum_{i=1}^{S} fitness_{1i}(T)} \tag{3.7}$$

$$P_i'(T)=\frac{fitness_{2i}(T)}{\sum_{i=1}^{S} fitness_{2i}(T)} \tag{3.8}$$

Where $fitness_i(T)$refers to the fitness of individual i; S stands for the scale of the population. Lastly, we conduct election of the individuals by rotating selected operation, as above mentioned, the individual with the high fitness has high probability to be selected and individuals with low fitness also have the chance to be chosen. To realize this, we program to get a rand number k, belong to (0,1). If there is a random number k satisfying $P_1+P_2+\ldots+P_{i-1} < k \leqslant P_1+P_2+\ldots+P_i$, the i^{th} individual will be selected. So, individuals with bigger fitness ratio have greater chance to be selected.

3.2.5 Adaptive Probabilities of Crossover and Mutation Operation

Crossover operation is to produce new individuals by substituting and reforming parts of the two subsequently selected parental individuals. Through hybridization the searching ability of genetic algorithm gets tremendous improvement. Each individual has a certain probability to cross with another individual. There are one-point crossover and multi-point crossover. This paper chooses one point crossover which randomly selects the cross point and exchange the sequence. Mutation operation is a local search mechanism which can effectively maintain the population diversity. It adjusts local optimal solution, develops new search solution space, and reflects the coverage to the optimal solution.

The probabilities of crossover(p_c) and mutation(p_m) greatly determine the degree of solution accuracy and the convergence speed that GA can obtain. Increasing values of P_c and P_m promote exploration at the expense of exploitation. Moderately large values of P_c(0.5-1.0) and small values of P_m(0.001-0.05) are commonly employed in GA practice. Genetic algorithm with adaptive parameters is significant and promising variant of genetic algorithms. Instead of using fixed values of p_c, p_m, we adopt adaptive probabilities of crossover and mutation [12]. A parameter setting of IGA is illustrated in Table 1.

$$P_c=\begin{cases} k_1(f_{max}-f')/(f_{max}-\bar{f}), & f'<\bar{f} \\ k_2, & f'<\bar{f} \end{cases} \tag{3.9}$$

$$P_m = \begin{cases} k_3(f_{max}-f')/(f_{max}-\bar{f}), & f'<\bar{f} \\ k_4, & f'<\bar{f} \end{cases} \tag{3.10}$$

Where $k_1, k_2, k_3, k_4 < 1.0$

IV. RESULT EVALUATION

In this section, we estimate the proposed SCM-cloud framework with novel algorithms implemented for security. This section comprises two sub-sections: simulation setup, and comparative analysis.

4.1 Simulation Setup

The proposed novel SCM-cloud architecture is implemented in the Java simulation environment. Table 2 defines the most important parameters used in the proposed algorithm simulations. We vary the number of users, cloud service provider and difference time to analyze the network performance. However, the parameters are not limited to this list.

Table 2

Simulation Parameters	
Number Of user	10
Number Of cloud Provider	3
Queue Type	Drop Tail
Buffer Capacity	3
Data Rate	100 Mbps
Send Interval	2 Seconds
Simulation time	30 Seconds

4.2 Performance Evaluation

In this section, the QoS efficiency is defined in terms of three important throughput parameters as well as end-to-end delay and packet loss. Network throughput is defined as the amount of data transmitted from user to cloud in a specified time period. The throughput, T_p, is calculated using the following formula:

$$T_p = \frac{NP_s}{T}$$

Here, T represents the time interval of transmission, NP_s denotes the number of packet to be transferred. Figure 3 shows that, the proposed SCM model gradual improve of throughput when simulation time is increased.

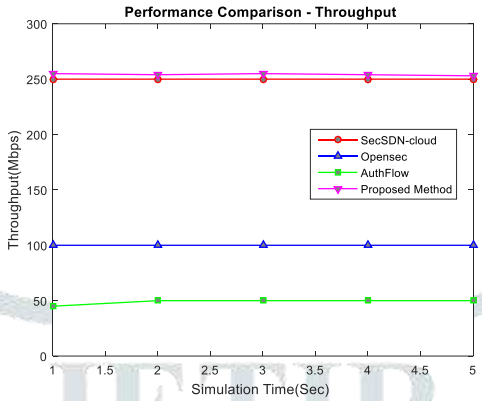

Figure 3: Performance Comparison of Throughput

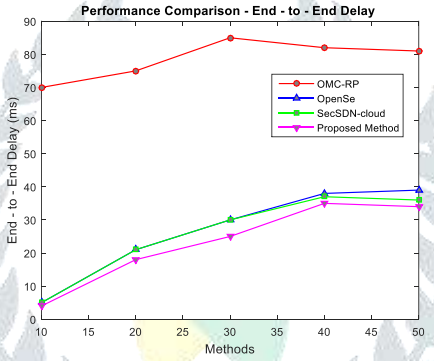

Figure 4: Performance Comparison of End-to-End Delay

Figure 4 shows the results attained for the end-to-end delay, which is defined as the time required to transmitting a packet from a user to a cloud server across the cloud environment. In the proposed SCM-cloud, this delay increases regularly as the simulation time increases because the number of incoming packets is increased. Delay happens because of authorization checking. Therefore, developments to these parameters are reflected by a reduction of the end-to-end delay metric. The proposed novel algorithms effectively achieve improvements in QoS.

Packet loss outcomes from packets discarded by a cloud server to receive the packet due to overloading. Because buffer and queue sizes are limited, packet overloading occurs when the system is unable to accept packets that surpass the existing level. Packets restricting from unauthorized data access can cause packet losses for legitimate users. Figure 5 show that the proposed SCM-cloud minimizes packet losses because it can mitigate unauthorized access.

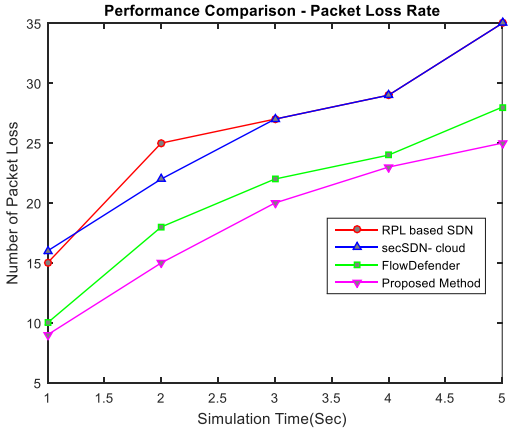

Figure 5: Performance Comparison of Packet Loss

Figure 6 and Figure 7 give the two performance metrics of algorithms. The x axis denotes the setting value of the deadline factor α. Due to α+β=1, we also know the setting value of β. From Figure 6 we can see that with the increment of the deadline factor, the normalized schedule cost of all four algorithms will reduce. This is because now the algorithms given more preference to the execution cost. In contrast, our algorithm proposed SCM can obtain the least NSC compared with other three algorithms. This means that the genetic mechanism has a better feasibility.

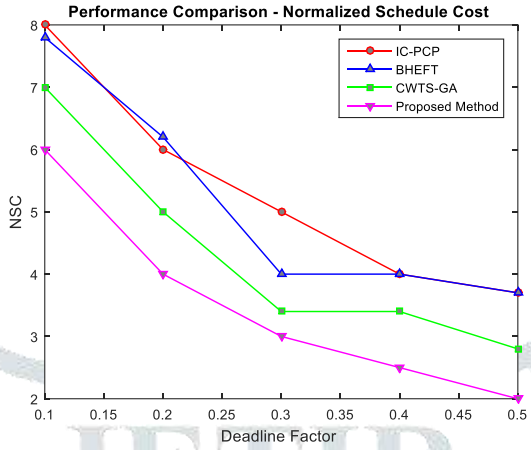

Figure 6: Performance Comparison of NSC

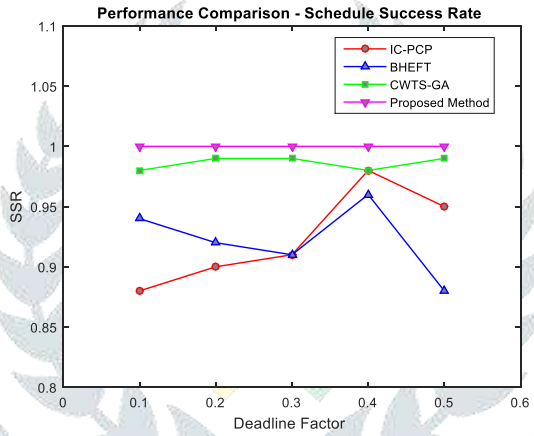

Figure 7: Performance Comparison of SSR

In Figure 7, we can see that proposed SCM model has the highest schedule success rate of workflow scheduling, which means this algorithm can better satisfy the constraints of the deadline and the budget and generate more feasible scheduling solutions in genetic population evolution. At the same time, due to the uncertain of the individual mutation, there are also some non-feasible scheduling solutions in our algorithm. But compared with other three algorithms, the schedule success rate has been improved by a large margin.

V. CONCLUSION

When organization or individuals utilize cloud storage services to share data, the most about problem is data security and privacy. A direct technique to defend the user's privacy is to encrypt the sensitive data before sending out. In this paper, we propose secure cloud architecture to defend against unauthorized access and improve the resource utilization. The prerequisite for user authentication is implemented through the DS-CSH algorithm, which uses SHA3-384. Only after authenticating a user do that user's data packets progress for analysis using the default policies. Furthermore, IGA algorithm to attain tasks scheduling with least make span and load balancing. Together, we apply greedy algorithm to initialize the population, rings in variance to illustrate the load intensive among nodes, weights multi-fitness function. Then we have experimentally evaluated performance of SCM in terms of throughput, Normalized schedule cost, Success rate, end to end delay and Number of Packet Loss. Thus, the proposed SCM provides a secure environment with a higher QoS that can support more users.

REFERENCES

[1] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energyaware server provisioning and load dispatching for connection-intensive internet services.," in NSDI, vol. 8, p. 337–350, 2008.

[2] Y. Jin and K. Li, "An optimal multimedia object allocation solution in multi-powermode storage systems," Concurrency and Computation: Practice and Experience, vol. 22, no. 13, pp. 852–1873, 2010.
[3] W. Lin, C. Liang, J. Z. Wang, and R. Buyya, "Bandwidth-aware divisible task scheduling for cloud computing," Software: Practice and Experience, vol. 44, no. 2, pp. 163–174, 2014.
[4] J. Gu, J. Hu, T. Zhao, and G. Sun, "A new resource scheduling strategy based on genetic algorithm in cloud computing environment.," Journal of Computers, vol. 7, no. 1, 2012.
[5] Wu F, Wu Q, Tan Y. Workflow scheduling in cloud: a survey[J]. Journal of Supercomputing, 2015, 71(9):3373-3418.
[6] J. Yu, R. Buyya, and K. Ramamohanarao, Workflow scheduling algorithms for grid computing[C], in Metaheuristics for scheduling in distributed computing environments. Springer, 2012:173-214.
[7] M. Maheswaran, S. Ali, H. J. Siegel, et.al, Dynamic mapping of a class of independent tasks onto heterogeneous computing systems [J], Journal of Parallel & Distributed Computing 2012,59(2) :107-131.
[8] A. Mendiola, J. Astorga, E. Jacob, and M. Higuero, "A survey on the contributions of Software-Defined Networking to Traffic Engineering," IEEE Communications Surveys & Tutorials, 2017.
[9] J. M. Wang, Y. Wang, X. Dai, and B. Bensaou, "Sdn-based multiclass qos guarantee in inter-data center communications," IEEE Transactions on Cloud Computing, 2015.
[10] J. Liu, Y. Li, H. Song, and D. Jin, "NetWatch: End-to-end Network Performance Measurement as a Service for Cloud," IEEE Transactions on Cloud Computing, 2016.
[11] K. Sood, S. Yu, Y. Xiang, and H. Cheng, "A General QoS Aware Flow-Balancing and Resource Management Scheme in Distributed Software-Defined Networks," IEEE access, vol. 4, pp. 7176-7185, 2016.
[12] K. Xu, H. Xiong, C. Wu, D. Stefan, and D. Yao, "Data-provenance verification for secure hosts," IEEE Transactions on Dependable and Secure Computing, vol. 9, pp. 173-183, 2012.
[13] M. H. Kabir, "Software Defined Networking (SDN): A Revolution in Computer Network," IOSR Journal of Computer Engineering (IOSR-JCE), vol. 15, pp. 103-106, 2013.
[14] C. Chen, X. Hu, K. Zheng, X. Wang, Y. Xiang, and J. Li, "HBD: towards efficient reactive rule dispatching in software-defined networks," Tsinghua Science and Technology, vol. 21, pp. 196-209, 2016.
[15] H. Jin, W. Dai, and D. Zou, "Theory and methodology of research on cloud security," Science China Information Sciences, vol. 59, pp. 050105:1-050105:3, 2016.
[16] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," IEEE Communications Surveys & Tutorials, vol. 18, pp. 602-622, 2016.
[17] T. Xu, D. Gao, P. Dong, H. Zhang, C. H. Foh, and H.-C. Chao, "Defending against new-flow attack in sdn-based internet of things," IEEE Access, vol. 5, pp. 3431-3443, 2017.
[18] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in software defined networks: A survey," IEEE Communications Surveys & Tutorials, vol. 17, pp. 2317-2346, 2015.
[19] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," Communications of the ACM, vol. 51, no. 1, pp. 107–113,2008.
[20] K. Chen and W.-M. Zheng, "Cloud computing: system instances and current research," Journal of Software, vol. 20, no. 5, pp. 1337–1348,2009.
[21] H. Kllapi, E. Sitaridi, M. M. Tsangaris, and Y. Ioannidis, "Schedule optimization for data processing flows on the cloud," in Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, pp. 289–300, ACM, 2011.
[22]M. D. de Assunc ̧ao, A. di Costanzo, and R. Buyya, "A cost-benefit ̃ analysis of using cloud computing to extend the capacity of clusters,"Cluster Computing, vol. 13, no. 3, pp. 335–347, 2010.
[23] W. Lin, C. Liang, J. Z. Wang, and R. Buyya, "Bandwidth-aware divisible task scheduling for cloud computing," Software: Practice and Experience, vol. 44, no. 2, pp. 163–174, 2014.
[24] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," Systems, Man and Cybernetics, IEEE Transactions on, vol. 24, no. 4, pp. 656–667, 1994.
[25] B. Pavez-Lazo and J. Soto-Cartes, "A deterministic annular crossover genetic algorithm optimisation for the unit commitment problem," Expert Systems with Applications, vol. 38, no. 6, pp. 6523–6529, 2011.
[26] S.-H. Ling and F. F. Leung, "An improved genetic algorithm with average-bound crossover and wavelet mutation operations," Soft Computing, vol. 11, no. 1, pp. 7–31, 2007.
[27] L. Wei and M. Zhao, "A niche hybrid genetic algorithm for global optimization of continuous multimodal functions," Applied Mathematics and Computation, vol. 160, no. 3, pp. 649–661, 2005.
[28] F. Li, L. D. Xu, C. Jin, and H. Wang, "Intelligent bionic genetic algorithm (ib-ga) and its convergence," Expert Systems with Applications, vol. 38, no. 7, pp. 8804–8811, 2011.
[29] R.-F. Wang, L.-C. Jiao, F. Liu, and S.-Y. Yang, "Nature computation with self-adaptive dynamic control strategy of population size," Ruanjian Xuebao/Journal of Software, vol. 23, no. 7, pp. 1760–1772, 2012.
[30] W.-m. Zou and J. Y, "Consumer satisfaction genetic algorithm in cloud computing," Application Research of Computers, vol. 31, no. 1, pp. 85– 88, 2014.
[31] J.-F. Li and J. Peng, "Task scheduling algorithm based on improved genetic algorithm in cloud computing environment," Jisuanji Yingyong/ Journal of Computer Applications, vol. 31, no. 1, pp. 184–186, 2011.
[32] K. Elmeleegy, "Piranha: Optimizing short jobs in hadoop," Proceedings of the VLDB Endowment, vol. 6, no. 11, pp. 985–996, 2013.