Tasks Selection Policies for Securing Sensitive Data on Workflow Scheduling in Clouds

Henrique Yoshikazu Shishido
[1]University of São Paulo
São Carlos, SP, Brazil
[2]Federal University of Technology of Paraná
Cornélio Procópio, PR, Brazil
shishido@{usp.br,utfpr.edu.br}

Júlio Cezar Estrella,
Claudio F. Motta Toledo
University of São Paulo
São Carlos, SP, Brazil
{jcezar,claudio}@icmc.usp.br

Stephan Reiff-Marganiec
Department of Computer Science,
University of Leicester,
Leicester, United Kingdom
srm13@le.ac.uk

***Abstract*—Scheduling is an important topic to support data security for workflow execution in clouds. Some workflow scheduling algorithms use security services such as authentication, integrity verification, and encryption for all workflow tasks. However, applying security services to no sensitive data does not make sense as no benefit is gained, yet it increases the makespan and monetary costs. In this paper, we introduce five policies for selection of tasks that handle sensitive data. We also propose a workflow scheduling algorithm based on a multi-populational genetic algorithm for minimizing cost while meeting a deadline. Experiments using four workflow applications show that our proposal can minimize both the makespan and cost while maintaining the security of sensitive data compared to another approach in the literature.**

***Index Terms*—workflow scheduling; cloud computing; security; cost; deadline; multi-population genetic algorithm (MPGA).**

I. Introduction

Workflows form a very typical application model, especially in scientific, engineering and business fields. Essentially a workflow is a Directed Acyclic Graph (DAG), where each node represents a task, and an edge is a task dependence [1]. Workflows face many issues including data security concern and distributed processing on heterogeneous resources such as cloud computing [2]. Despite the many benefits of clouds, users or organizations are often reluctant to adopt cloud computing due to security reasons [3].

Workflow scheduling is a topic vastly investigated in cloud computing. Recently, workflow scheduling algorithms have also focused on security, e.g., [4]–[10]. Workflow scheduling algorithms based on security constraint consider either setting up private clouds for processing sensitive data and public clouds to the non-sensitive ones or apply authentication, integrity verification and encryption on all data processed in public clouds.

Li et al. [8] and Arunarani et al. [11] presented algorithms to find schedules that combines different levels of authentication, integrity verification, and encryption services, and a VM instance for each task for minimizing the execution cost, while meeting the minimum security requirement and deadline. However, these algorithms do not guarantee that all tasks that handle sensitive data (critical tasks) will be secured, due the codification of the optimization. For instance, consider a workflow composed by four tasks A-B-C-D, where A-B-C handle non-sensitive data, and D is the critical task. Consider solutions s_1 which applies security services for tasks A-B-C, and leaves D with no security and s_2 where A-C-D are secured, while B is not. The algorithms in [8] and [11] do not take into consideration the semantics of each workflow task. In their work, both solutions s_1 and s_2 have secured three tasks and, therefore, present the same security level. However, solution s_1 suffers from a security breach leaving the critical task D with no protection, while solution s_2 protect critical task D, but wasting processing time securing B and C.

To address the need for securing relevant tasks while not wasting resources, we introduce the concept of task selection policies for securing workflow tasks that process sensitive data in this paper. We propose five policies that can be selected by the user in the workflow modeling phase, namely $entry/end$, $gather$, $scatter$, $gather/scatter$, and all. The novelty of our work is the (1) introduction of the five task selection policies that can be chosen by the user according to the data sensitivity of the workflow tasks and (2) a scheduling algorithm supporting the policies. Unlike previous work in the literature that applies security services to all workflow data, security services are then automatically and exclusively applied to the tasks covered by the policy.

II. System models and problem formulation

A workflow is commonly structured as a Directed Acyclic Graph (DAG) $w = (T, E)$, where $T = \{t_0, t_1, ..., t_i, ..., t_{n-1}\}$ is a set of tasks, and E is a set of edges $e(i, j)$ that represents the dependencies among task t_i and t_j. For edge $e_{i,j} \in E$, t_i is an immediate predecessor task of task t_j and t_j is an immediate successor of task t_i. $pred(t_i)$ denotes a set of all the immediate predecessors of t_i.

The cloud datacenter model considers a set of instance series $S = (s_0, s_1, s_2)$. Each series s_i contains a set of VM types $VM_{s_i} = \{VM^1_{s_i}, ..., VM^2_{s_i}, ..., vm^k_{s_i}\}$. A VM type $vm^k_{s_i}$ is indicated by characteristics as instance series type s, number of virtual processors $vcpu^k_{s_i}$, processing capacity in MFlops $p^k_{s_i}$, memory $m^k_{s_i}$ and storage st^k_s, and the monetary cost $c^k_{s_i}$.

It is assumed that all virtual machines have the same data communication bandwidth and the transmission cost among them is zero.

The security model sets three types of security services $sl_i = [sl_i^a, sl_i^g, sl_i^c]$, where each task t_i is protected by an authentication service sl_i^a, integrity verification service sl_i^g, and encryption service sl_i^c. Security services increase the overhead of the workflow execution. Eq. 1 defines the overhead produced by integrity and encryption services, where d_i^l is the amount of data to be protected using a service level sl_i^l.

$$SC^l(t_i) = F^l(sl_i^l, d_i^l), \qquad l \in \{g, c\} \tag{1}$$

The authentication service produces an overhead of constant time, independent of the amount of data. Eq. 2 denotes the overhead of authentication service.

$$SC^l(t_i) = F^l(sl_i^l), \qquad l \in \{a\} \tag{2}$$

The total overhead involving all three types of security services is a simple sum of the overheads.

One algorithm for each type of security service sl_i^l was considered in our study, and there are different VM images to provide each security service. The transferring time from t_{i-1} to t_i is computed by $TT(t_i) = I_i/B$, where I_i is the size of data to transfer and B represents the network bandwidth. It is assumed that the size of the files is already known. When the transferring occurs between tasks allocated on the same VM, no time is computed.

The execution time of task t_i is calculated by $ET(t_i, vm_s^k) = W_i/p_s^k$, where W_i represents the task t_i workload and p_s^k is the processing capacity of the VM assigned to execute task t_i. After task processing, output data is submitted to an encryption service to minimize the risk of unauthorized interception. Finally, the total processing time $PT(t_i, vm_s^k)$ of a task t_i is the sum of the transferring time $TT(t_i)$, task execution time $ET(t_i, vm_s^k)$ and the overhead with security services $SC(t_i)$.

A. Problem formulation

The addressed problem is to find a schedule to execute a workflow in the public cloud optimizing execution cost, respecting deadline constraints and offering safer processing of critical tasks. The scheduling scheme is denoted by $schedule = (R, M, TEC, TET)$ in terms of resources (R), task-resource mapping (M), total execution cost (TEC), and total execution time (TET).

$R = \{r_0, r_1, ..., r_{n-1}\}$ is the set of provisioned VMs for workflows' tasks, where r_i is a three tuples of form $r_i = (vm_s^k(t_i), LST(t_i, vm_s^k), LET(t_i, vm_s^k))$. $vm_s^k(t_i)$ is the VM type leased to a task t_i, which has a lease start time $LST(t_i, vm_s^k)$ and a lease end time $LET(t_i, vm_s^k))$. M is the mapping of a task to a VM $m(t_i, vm_s^k) = (t_i, vm_s^k(t_i), ST(t_i), ET(t_i))$ in terms t_i is the task mapped to vm_s^k that is expected to start at $ST(t_i)$ and complete at time $ET(t_i)$. The remaining two terms total execution cost TEC and total execution time TET are denoted by $TEC = \sum_{i=0}^{n-1} c_s^k \cdot \lceil LET(t_i, vm_s^k) - LST(t_i, vm_s^k) \rceil$ and $TET = max\{ET(t_i) | t_i \in T\}$. The term $c_s^k \cdot \lceil LET(t_i, vm_s^k) - LST(t_i, vm_s^k) \rceil$ is the execution cost for task t_i on vm_s^k. Formally, the problem is defined as: $Minimize: TEC \; and \; Risk(w) subject \; to: TET \leq T_D$

III. Method

A. Selection of critical tasks

A workflow can present different tasks types as an entry, end, gathering, scattering, and both gathering and scattering. Each task type performs specific activities and has implications if its data are stolen. We propose five task selection policies that can be chosen according to the security requirements as shown in Figure 1. The first policy is the $Entry/End$ policy which assures securing all data handled in both entry and end tasks. It can be chosen in workflows where the initial and final data must be protected. $Gather$ and $Scatter$ policies offer protection for tasks that receive sensitive data from multiple tasks or send sensitive data to multiple tasks, respectively. $Gather/Scatter$ secure the tasks which have multiple parent tasks and multiple child tasks. Finally, All policy is used in cases that all tasks' data need a secured treatment.

All five policies apply authentication, integrity verification, and encryption services for securing critical tasks. The policies could be offered by the workflow management system (WfMS) as an option for securing the workflow execution. The proposed approach requires the knowledge of semantics of each workflow task.

B. Coding strategy and scheduling generation

We used a genetic-based algorithm to minimize the execution cost and meet a deadline as stated in [12]. Our strategy discards the security services into the chromosome codification, avoiding the possibility that critical tasks do not receive adequate safety treatment. Critical tasks will be those that fit the policy chosen by the user. Each critical task will adopt a unique level for each type of security service.

The fitness is given by $fitness = TEC \cdot Risk(w)$, if $TET \leq T_D$, where TEC is the total execution cost, $Risk(w)$ is the workflow execution risk, TET is the total execution time and the $deadline$ is the maximum execution time acceptable. Otherwise, a penalty $fitness = TEC \cdot 10^6$ is applied to.

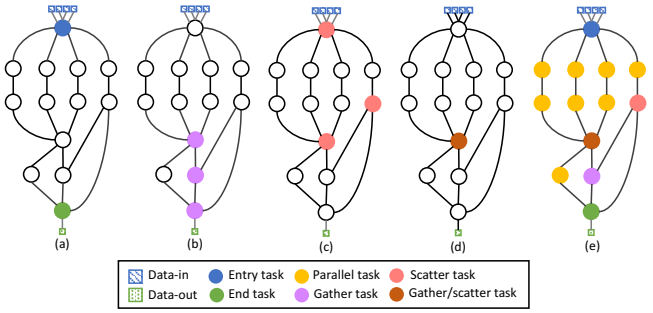

Fig. 1. Task selection policies: (a) Entry and End; (b) Gather; (c) Scatter; (d) Gather/scatter; (e) All.

IV. Results

In this section, the performance evaluation of WS-TSP is presented. Our experiments were conducted using WorkflowSim [13] as a simulation tool. Four real-world workflows from different scientific areas were used, namely CyberShake, Epigenomics, Montage, and SIPHT. We analyzed the five policies regarding security risk and cost. We compared WS-TSP to an adapted SCAS approach [8]. We also included the No policy where no task is protected. It was supposed that VMs are allocated in the same datacenter and connected through a 1 Gbit network. Amazon EC2 instance specifications and prices [14] were considered for experimentation.

The execution deadline constraint T_D was defined by the average between the fastest and slowest execution time. The former was the execution considering the cheapest VM type (slowest), while the second one consists of execution on expensive VM type (fastest). Both executions considered applying security services to all workflow's tasks. Thus, there is always a solution that can satisfy the deadline constraint. MPGA parameters were set empirically based on previous tests [12] with three populations, 50 individuals per population, 5.0 of crossover rate, 0.7 of mutation rate, and two ramifications. We considered time as stop criterion (900 seconds).

For computing the security cost SC_{ti}, CBC_MAC_AES, TIGER, and IDEA as authentication, integrity verification and encryption services were considered, respectively. It was assumed that CBC_MAC_AES, TIGER and IDEA demand 163ms/auth, 4.36 kB/ms and 13.50 kB/ms, respectively.

A. Comparison of makespan and cost

The cost and makespan for the five task selection policies were analyzed. Fastest VMs get the shortest makespan for all workflows and task selection policies as expected (Figure 2). However, the execution cost exceeds $1500. On the other hand, the executions with only slowest VMs require a lesser budget but carry on long makespan. The reduction of makespan in CyberShake and Montage workflows is not significant for executions using No policy on fastest VM instances. It is explained due both two workflows concentrate the most CPU utilization on few tasks. Thus, faster instances on all tasks do not ensure a better makespan. Epigenomics presents higher makespan under slowest VMs because it has more CPU-intensive tasks than Cybershake and Montage. All policy applies security services to all workflow's tasks and consequently, produces the highest makespan on all workflows. This policy presents the most utilization of VMs and is the most benefitted by faster VMs to process the security services.

The $Gather$ policy increased the overhead on Montage, Epigenomics, and SIPHT, but not CyberShake workflow. The reason is that CyberShake has two gather tasks, but both involve little data to be secured, resulting in low overhead. The $Scatter$ policy has impacted on CyberShake because each task on the second level manipulates a large amount of data (approx. 547 MB). Epigenomics has only one scatter task that reads 242 MB of input and splits into multiple output files. Security services increased slightly the overhead due its unique scatter task that process 48 MB of input data. Considering all four workflows, only SIPHT has a gather/scatter task that extended the makespan slightly. $Entry\_End$ and All policies increased the makespan on CyberShake, possibly due the $Entry\_End$ tasks handle most of all workflow's data. $Entry\_End$ also prolonged makespan of SIPHT execution.

B. Risk comparison

We assumed that user knows the semantics of the workflow tasks to select the policy that covers the sensitive data. SCAS codifies four attributes per task: VM instance type, authentication, integrity verification and encryption services. To compare WS-TSP to SCAS, we have simplified SCAS codification to two attributes (VM instance and security activation) for each task. The security activation attribute corresponds to the three security services (authentication, integrity verification, and encryption). It can assume values 0 and 1, where 0 sets no security services to the task, and 1 applies the three security services to the task.

Security risk was established to consider the percentual risk $Risk(t_i)$ of each critical task t_i on the whole workflow risk as defined by Equation 3. The $pred(t_i)$ and $succ(t_i)$ represent the total of parent and child tasks of a critical task t_i, and $sensitive\_edges(w)$ is the sum of the edges of all tasks that handle sensitive data. The overall workflow risk $Risk(w)$ is obtained from the sum of percentual tasks risk $Risk(t_i)$ as shown in Eq. 4. The sum considers only the tasks where its security activation attribute is equal 1. The results show that WS-TSP was able to handle all tasks involving sensitive data as shown in Figure 3. This is because the prior knowledge of critical tasks allows the user to select a policy that ensures them. SCAS exposed the workflow's sensitive data to security risks. We can note that task data $gather$, $scatter$, $entry/end$, all were partially vulnerable to cyber-attacks. CyberShake, Montage, and Epigenomics workflows do not have $gather/scatter$ tasks; then the security risk is 0.

$$Risk(t_i) = \frac{pred(t_i) + succ(t_i)}{sensitive\_edges(w)} \tag{3}$$

$$Risk(w) = 1 - \sum_{i=1}^{nSec} Risk(t_i) * sec(t_i) \tag{4}$$

V. Concluding remarks

We proposed the WS-TSP approach composed of five task selection policies and a genetic-based scheduling algorithm which takes advantage from tasks semantics for minimizing workflow execution cost, preserving the privacy of critical tasks while respecting the deadline. The results indicate that task semantics can contribute to minimizing the execution cost and data leakage. When compared to SCAS approach, WS-TSP achieved significant cost minimization with a slight increase of the makespan. A limitation of WS-TSP is that the user must know the task semantics to choose an appropriate policy. The evaluation using four workflows does not allow us to say that WS-TSP could be efficient with any workflow, so further case studies could be explored.

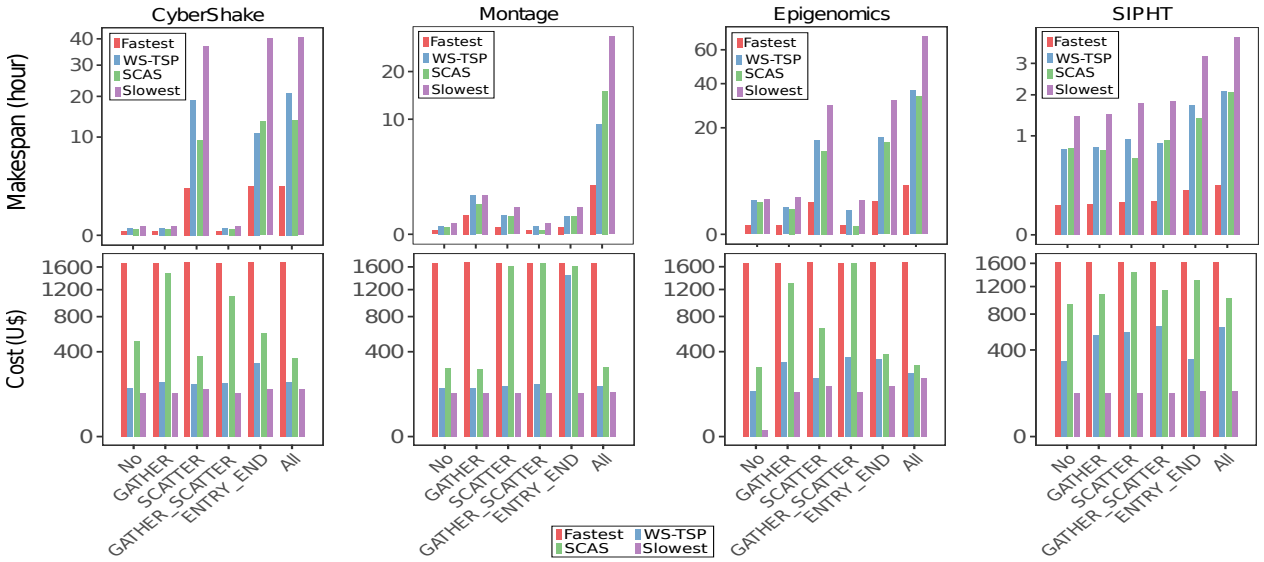

Fig. 2. Cost and makespan of different task selection policies and workflows using Fastest, Slowest, SCAS and WS-TSP.

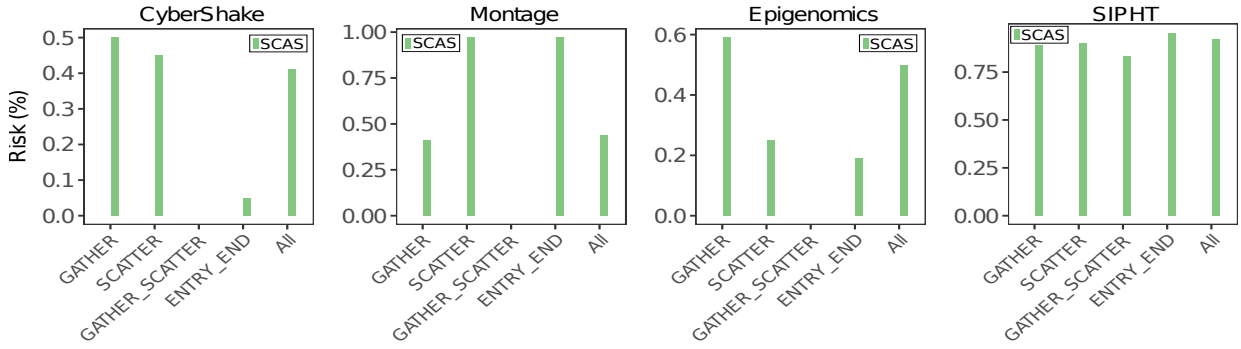

Fig. 3. Percentage of execution risk for the different task selection policies and workflows, considering SCAS.

ACKNOWLEDGMENTS

The authors acknowledge CAPES, FAPESP, CNPq and USP for the resources provided, USP and UTFPR for the scholarship to Henrique Yoshikazu Shishido.

REFERENCES

[1] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-science: An overview of workflow system features and capabilities," *Future Gener Comput Syst*, vol. 25, no. 5, pp. 528 – 540, 2009.

[2] R. Barga and D. Gannon, *Scientific versus Business Workflows*. Springer, 2007, pp. 9–16.

[3] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," *J of Internet Serv App*, vol. 4, no. 1, pp. 1–13, 2013.

[4] P. Watson, "A multi-level security model for partitioning workflows over federated clouds," *J Cloud Comp: Adv Syst App*, vol. 1, no. 1, p. 15, 2012.

[5] S. Sharif, J. Taheri, A. Y. Zomaya, and S. Nepal, "Mphc: Preserving privacy for workflow execution in hybrid clouds," in *Int Conf Parallel and Distrib Comp App Tech*, 2013, pp. 272–280.

[6] H. Liu, A. Abraham, V. Snášel, and S. McLoone, "Swarm scheduling approaches for workflow applications with security constraints in distributed data-intensive computing environments," *Inf Sci*, vol. 192, pp. 228–243, 2012.

[7] L. Zeng, B. Veeravalli, and X. Li, "Saba: A security-aware and budget-aware workflow scheduling strategy in clouds," *J Parallel Distrib Comp*, vol. 75, pp. 141 – 151, 2015.

[8] Z. Li, J. Ge, H. Yang, L. Huang, H. Hu, H. Hu, and B. Luo, "A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds," *Future Gener Comput Syst*, vol. 65, pp. 140 – 152, 2016.

[9] X. Zhu, Y. Zha, P. Jiao, and H. Chen, "Security-aware workflow scheduling with selective task duplication in clouds," in *Proc High Perform Comput Symp*, 2016, pp. 1–8.

[10] C. Jianfang, C. Junjie, and Z. Qingshan, "An optimized scheduling algorithm on cloud workflow using discrete particle swarm," *Cybern Inf Tech*, vol. 14, no. 1, pp. 25–39, 2014.

[11] A. R. Arunarani, D. Manjula, and V. Sugumaran, "Ffbat: A security and cost-aware workflow scheduling approach combining firefly and bat algorithms," *Concurr Comput*, vol. 29, no. 24, pp. e4295–n/a, 2017.

[12] H. Y. Shishido, J. C. Estrella, C. F. M. Toledo, and M. S. Arantes, "Genetic-based algorithms applied to workflow scheduling algorithm with security and deadline constraints in clouds," *Comput Electr Eng*, 2017.

[13] W. Chen and E. Deelman, "Workflowsim: A toolkit for simulating scientific workflows in distributed environments," in *Proc IEEE Int Conf on e-Science*. IEEE, 2012, pp. 1–8.

[14] A. EC2, "Amazon EC2 Pricing ," https://aws.amazon.com/ec2/pricing, 2018, [Online; accessed 12-Dec-2017].