A CloudSim extension for evaluating security overhead in workflow execution 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*—Workflow scheduling algorithms for cloud environments are extensively studied using workflow management system simulators. The common criteria covered by algorithms and addressed in simulators are makespan, monetary cost, reliability, and energy consumption. Beyond these criteria, security is also a criterion and has been investigated recently. Scientific and business workflows typically handle sensitive and big data that can influence the makespan and cost significantly when a scheduling algorithm applies security services to these data. However, simulators for workflow execution do not address the overhead produced by applying security services to sensitive data. In this paper, we propose an extension for workflow simulator to support security services. We considered seven steps for measuring security overheads on workflow execution. The extension was validated by executing a real-world workflow applying three types of security services namely authentication, integrity verification, and encryption. Our extension proved useful for simulating workflow execution applying security services on sensitive data and analyzing the effects of security on makespan, cost and security criteria.**

***Index Terms*—simulator,security,overhead,workflow,scheduling**

I. Introduction

Cloud computing is a distributed paradigm that delivers on-demand computing resources over the internet on a pay-for-use basis. The three primary service models in clouds are Software (SaaS), platform (PaaS), and infrastructure as a service (IaaS). The latter provides computing resources (servers, storage, and networking) so that organizations do not need to invest in local infrastructure, high-performance hardware, and human resources for its management. IaaS also allows the scaling of infrastructure for supporting dynamic workloads [1]. Despite the advantages of using IaaS, some users are resisting its adoption, with security one of the main reasons given [2].

Many areas have become data-driven, and new scientific and business knowledge is found by putting together data analysis and knowledge discovery pipelines using a workflow application model [3]. Directed Acyclic Graph (DAG) is commonly used to represent workflows, where each node represents a task, and an edge is task dependence [4]. Workflows can be classified as either business or scientific. Business workflows are developed by business analysts, while the scientific ones typically involve more data and are modeled by scientists. Both business and scientific workflows face issues including the execution in distributed and heterogeneous resources and security [5]. Scientific workflows tend to take longer makespan due to having to apply security services to big data.

Workflow scheduling in a cloud environment has been studied considering different criteria (makespan, cost, energy, and reliability) [6]. Security is a criterion more recently addressed in scheduling algorithms when workflows are executed in cloud environment [7]–[13]. Scheduling algorithms are commonly developed using simulated experiments. Workflow simulators allow users to determine the correctness and efficiency of a scheduling algorithm before the algorithm is deployed in real systems. Another benefit of workflow simulators is that they permit to study a problem at different levels of abstraction with no money spent on renting cloud infrastructure. The simulation tools commonly used for workflow execution are based on the CloudSim toolkit [14] such as WorkflowSim [15], DynamicCloudSim [16] and SimpleWorkflow. These simulators can (a) parse workflow description files; (b) consider different computing resources; (c) change scheduling algorithms, and; (d) show execution metrics like cost and makespan. However, security is a fundamental aspect that is not addressed by these simulators. Neglecting the overhead resulting from applying security services to big data workflows may produce incorrect makespan and cost in the simulation of workflow executions.

In this paper, we (1) introduce an extension for simulations to consider security aspects in workflow execution in clouds. We have introduced a model that can receive a scheduling encoding containing VM types and security parameters for each task to compute the cost, makespan, and risk of success from malicious attacks. We also (2) implemented our extension in the WorkflowSim tool and validate it using a real-world workflow. To the best of our knowledge, the extension presented here provides one of the first studies into how to simulate the overhead applying security services in workflow execution in a cloud. Nevertheless, this study does not cover security approaches using hybrid clouds, where sensitive tasks are placed in private clouds, while the non-sensitive tasks are executed in public clouds.

The remainder of the paper proceeds as follows: security issues in clouds are presented in Section II. A case study of a workflow simulator is shown in Section III. The requirements for workflow simulation considering security services are discussed in Section IV. The case study of a framework based on security simulation and its usage example are presented in Sections V and VI. Finally, Section VII establishes the concluding remarks and future directions of this research.

II. Security of workflow execution in clouds

The present study relies on the applying of security services to sensitive tasks as a protection measure against malicious attacks in workflow execution in clouds. Security is defined herein as a quantitative assessment of the level of protection afforded to tasks that handle sensitive data in a workflow applying security services on sensitive tasks. A sensitive task is defined as a workflow's task which has input or output data that should not be made public and should only be disclosed under limited circumstances. Users must be authorized to access sensitive data since the unauthorized disclosure, alteration or destruction may cause perceptible damage to the institution.

There are several ways to protect sensitive data from malicious attacks. In this study, we consider protecting them through adopting security services for authentication, integrity verification, and encryption. *Authentication* services are applied to avoid unauthorized access or disclosure when the task execution involves the transmission of data over a network. It is the process of confirming the identification of an entity (user, systems, machines, etc.) that is attempting to access resources. Authentication is often confused with authorization. While the authentication process verifies an identity, authorization verifies that the entity has the right permissions to access the requested resource.

Public clouds are characterized by multi-tenancy so that different users can be allocated in the same resource. This scenario creates conditions for tampering attacks, in which a malicious user could modify the data due to network issues. It is interesting that security systems use *integrity verification* services to ensure the accuracy and consistency of data transmitted. In workflow execution, an integrity service has the same intent, which is to certify that a data from one task is received by another one precisely equal it was sent.

The processing of a workflow task which involves data must use a readable format by the application. However, the transmission of sensitive data from a task to another one cannot be performed using a readable format. Therefore, *encrypting* data using encryption algorithms reduces the risk of data disclosure in a multi-tenant environment. Encrypted data is also called cipher-text, while unencrypted data is known as plain-text. Encryption algorithms typically translate plain-text into cipher-text so that only users with access to a decryption key or password can read it. There are two types of data encryption algorithms: asymmetric and symmetric encryption. Symmetric-key algorithms use the same secret key for encrypting and decrypting a data. Symmetric-key algorithms are faster than asymmetric encryption, but the sender-side must exchange the encryption key with the recipient before the decryption process. Asymmetric cryptography, also called as public-key cryptography, uses one public and one private key. The public key may be shared with everyone, while the private one must be protected. Anyone can encrypt a message using the recipient's public key. However, the message in question can be only decrypted by the recipient's private key. This characteristic makes asymmetric cryptography also accomplish an authentication function.

These three types of security services are considered in our execution model for sensitive data protection, but there are other types of security services that can support data protection which can easily be added to the simulation model.

III. Case study: WorkflowSim

The performance evaluation of workflow scheduling algorithms in real infrastructure is complex, time consuming and costly. As a consequence, simulation-based experiments have been consolidated as an acceptable way to evaluate workflow systems. Simulations reduce the complexity of the experimental setup, save effort in workflow execution, and enable a controlled environment for reproduction of experiments.

In the midst of workflow system simulators, WorkflowSim [15] is a tool developed at the University of Southern California, which extends the CloudSim toolkit by providing a layer of workflow management. WorkflowSim follows CloudSim's core engine, which iterates over a future-event list. Each *event* is related to a simulation *entity*. WorkflowSim extracts the typical features present in various workflow management systems (WMS) for handling workflow execution. WorkflowSim also covers task scheduling and execution overheads. The model adopted in WorkflowSim is based on the Pegasus workflow management system [17] as shown in Figure 1. It contains a *Workflow Mapper* for mapping abstract workflows to concrete workflows that are dependent on execution sites; a *Workflow Engine* to parse the data dependencies, and; a *Workflow Scheduler* to associate tasks to computing resources.

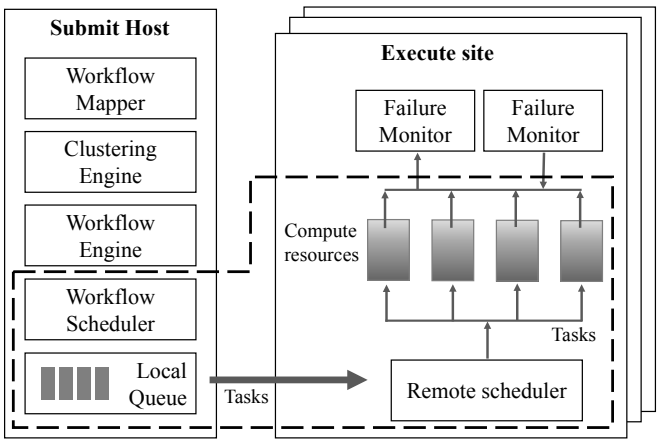

Fig. 1. WorkflowSim overview. The area surrounded by dashed lines is supported by CloudSim [15].

Listing 1. Simplified example of DAX file.

```
<job id="ID0001" name="parseData" length="385928">
  <uses file="initial.zip" link="input" size="105832"/>
  <uses file="a.zip" link="output" size="43659"/>
</job>
<job id="ID0002" name="processing"  length="1388">
  <uses file="a.zip" link="input" size="43659"/>
  <uses file="b.zip" link="output" size="15659"/>
</job>
<job id="ID0003" name="storeResults"  runtime="64851">
  <uses file="b.zip" link="input" size="15659"/>
  <uses file="c.zip" link="output" size="257271"/>
</job>
<child ref="ID0002>
  <parent ref="ID0001>
</child>
<child ref="ID0003>
   <parent ref="ID0002">
</child>
```

Workflow Mapper

Workflows can be represented using Directed Acyclic Graphs (DAGs), where each node represents a task to be computed, and edges are the data flow between jobs. The Workflow Mapper component is responsible for importing DAG files in XML format as shown in Listing 1. Each task is represented by tag **<job>** containing task identification id, name of task $name$ and task processing $length$ attributes. Moreover, tag **<uses>** represent which are the input and output data of a task. Tag **<child>** describes the dependency of each task that can contain one or more parent tasks represented by tag **<parent>**, except the entry tasks of the workflow. After mapping, the Workflow Mapper builds a task list and associates these tasks to an execution site.

Workflow Engine

The Workflow Engine controls the execution flow based on task dependencies to assure that a task can only be executed if all parent tasks have finished successfully. The Workflow Engine works together with the Workflow Scheduler, only releasing ready tasks to the scheduler.

Workflow Scheduler and Job Execution

According to tasks marked ready by the Workflow Engine, the Workflow Scheduler associates tasks to virtual machines based on the scheduling algorithm selected. WorkflowSim considers static (offline) and dynamic (online) scheduling algorithms. For the former, tasks are assigned to a VM instance before the workflow execution. In dynamic scheduling, tasks are associated with virtual machines according to their idleness during workflow execution. The essential difference between these two scheduling approaches, it is that static scheduling has already assumed the order and complexity of tasks at the start, while dynamic scheduling is used where resources can change due to run-time load and availability.

WorkflowSim introduced different layers of overheads and failures, which improves the accuracy of the simulation. However, security overhead feature still is not present in either WorkflowSim nor the other cited simulators.

IV. Requirements for security simulation

The security of sensitive data can be supported using authentication, integrity verification, and encryption algorithms. These algorithms can be available as security services and applied to workflow tasks for protecting sensitive data against malicious attacks in clouds. In this section, we describe the requirements to enable the evaluation of applying security services in workflow execution. A proposal of adding security overheads in the simulation of workflow execution is presented in Figure 2.

The first step concerns how to describe a workflow file appropriately. It is necessary to model a workflow considering the identification, instruction length and dependencies of each task. In the next step, the workflow must be parsed setting up the tasks and dependencies for controlling the workflow execution. The security overheads can be addressed by identifying which tasks handle sensitive data and which security services will be applied. To identify sensitive tasks, we propose an encoding that provides an extensible way to represent which tasks require security attention setting up security services as shown in Figure 3. Each task is protected by a set of security services including authentication, integrity verification, and encryption algorithms. Several algorithms offer different levels of protection and lead to different overheads. Algorithms with higher security levels demand more time to process the same amount of data. We propose a **SecurityAlgorithms** entity containing various algorithms for adapting the security requirement of each task. Associating different levels of security according to the task security requirements can produce less overhead and, consequently, reduce the makespan and the cost of a workflow execution. The process of task execution applying security services is shown in Figure 4. First, task t_i needs to be authenticated for transferring input data from the predecessor(s) task(s). After the transferring of all input data is complete, an integrity verification is performed. If the data has not been tampered with, the task execution can proceed. At the end, the output data are submitted to an encryption service before it will be sent to the child task(s).

The encoding array enc showed in Figure 3 also defines the VM instance type for each task t_i in $pos[4*i]$. Clouds provide a variety of heterogeneous computing resources, and each one is characterized by different CPU capacity, memory, storage, network bandwidth, cost, etc. We propose the **VmInstances**

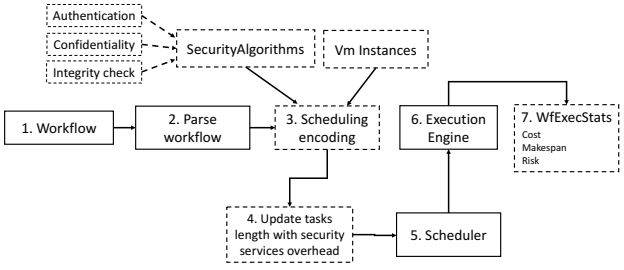

Fig. 2. Simulation of workflow execution adding security overhead.

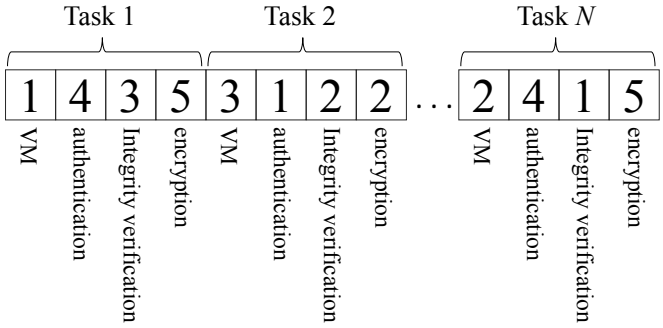

Fig. 3. Example of scheduling encoding using meta-heuristic optimization.

entity for optimization purposes since security algorithms take different time for executing in different VM instances. Using this encoding allows optimizing the workflow scheduling using meta-heuristic algorithms. The levels of authentication, integrity verification, and encryption services for a task t_i are defined in $pos[4 * i + 1]$, $pos[4 * i + 2]$, and $pos[4 * i + 3]$, respectively. This codification considers that higher values correspond to faster VM instances or safer security services.

In our model, we propose to add the security overhead into the task length. We assumed a, g, c to represent authentication, integrity verification and encryption, respectively. The authentication service takes a constant time, while integrity verification and encryption services are data size dependent. Simulators of workflow management systems typically represent the task complexity by using the million instructions (MI) metric. The security overhead must be represented in the same manner. Each security service is measured using different metrics. Authentication is measured in milliseconds (ms), while integrity verification and encryption services are based on kilobytes per second (kB/s). Eq. 1 and 2 show the function to convert the security length SL^{β} applying security services on task t_i, where $data\_size$ is the amount of data submitted to a security service β, $secAlg_{\beta}$ represents the throughput of a security service, and $cpu\_capacity$ corresponds to the CPU speed in MI. After SL^{β} computed for the three security services, they are added to task t_i length.

$$SL^{\beta}(t_i) = secAlg_{\beta} * cpu\_capacity, \qquad \beta \in \{a\} \quad (1)$$

$$SL^{\beta}(t_i) = \frac{data\_size}{secAlg_{\beta}} * cpu\_capacity, \qquad \beta \in \{g, c\} \quad (2)$$

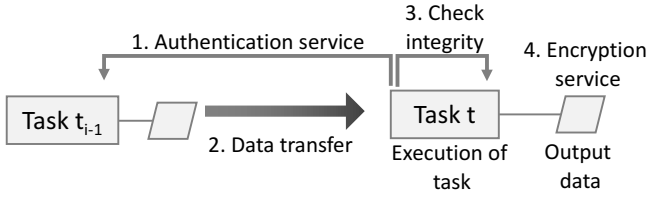

Fig. 4. Task execution process applying security services.

In the scheduling phase, all workflow tasks are assigned to available compute resources. If a scheduling encoding was previously specified in the initialization of simulation, the scheduler is skipped. Finally, all tasks are submitted to the workflow engine which controls the execution flow of tasks. Finally, the workflow execution metrics as cost, makespan, and security metrics are displayed to the user.

V. Case study: WfSecurity Extension

The proposed extension for workflow simulation concerning security overhead integrates into the design of the WorkflowSim toolkit (version 1.0). Figure 5 depicts the class diagram of our work, called WfSec. The blue classes already exist in WorkflowSim, and the white ones are proposed for the extension. The class **`TaskSec`** extends the original **`Task`** class using three members: **`VM_Instances`**, **`Security_Algorithms`** and **`Encoding`**, which are required for computing the overhead.

Class **`TaskSec`** receives the workflows tasks and its dependencies from the workflow DAX file. This scheduling is supported by the **`Encoding`** interface, which receives the scheduling map represented in the **`schedEncoding`** array as shown in Figure 3. Representing the scheduling map as an array is useful for scheduling optimization where meta-heuristics algorithms can tune VM instances and security parameters.

Class **`TaskSec`** defines an attribute **`security_length`**, which is summed to the task processing length. The input data **`dataIn`** and output data **`dataOut`** are submitted to integrity verification service and encryption services, respectively, as previously shown in Figure 2. The security requirements for each task are set for computing the QoS met. Each task computes the security risk using the implementation of **`computeRisk()`** method based on the security provided in the scheduling process and the security requirements of a task. This method uses **`Security_Algorithms`** class to get the security levels of authentication, integrity verification, and encryption algorithms.

After reading the VM instances and security parameters of each task, the essential step can be performed. The security overhead is calculated by (Eq. 1 and Eq. 2) and embedded to the attribute **`length`** of task. At the end of the workflow execution, the simulator shows a summary of the makespan, cost, and the security metrics computed by calling **`printSecurity()`**, which is implemented computing each tasks risk value according to the security model defined.

VI. Usage example

This extension is available on GitHub[1]. As part of the ongoing research carried out by the authors' group, the workflow management system simulator is employed in a study on security- and cost aware workflow scheduling algorithms in three publications [13], [18], [19]. The studies investigate the application of meta-heuristic algorithms for optimizing the

[1] git://redmine.lasdpc.icmc.usp.br/lasdpc/wfsim_security.git

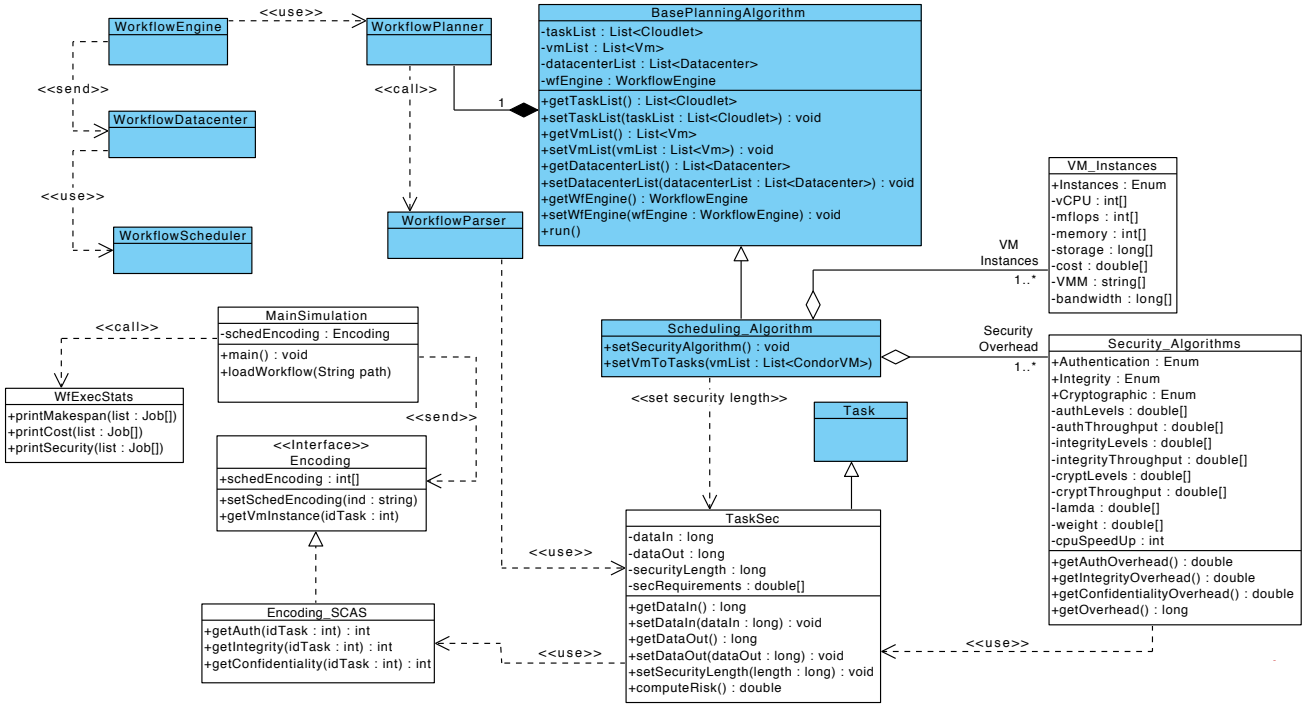

Fig. 5. Class diagram of WfSecurity extension. Classes in blue color represents WorkflowSim core, while the white ones represents WfSec extension.

assignment of different VM instances and security services for each task. In a first stage, we designed this extension for WorkflowSim aimed to evaluate scheduling algorithms that preserve the security of sensitive data.

In the WorkflowSim with WfSec extension, we implemented the **`MainSimulation`** class. Initially, we set the scheduling map which implements the **`Encoding`** interface following the encoding presented in Figure 3. The **`Security_Algorithms`** class was implemented by adding the list of algorithms and its corresponding throughput and security level as shown in Table I. Some encryption or hashing algorithms as AES and SHA256 are not absent due it was not benchmarked in the study that we followed. Also, the **`VM_Instances`** class was implemented based on Amazon EC2 instances and pricing model [20].

We considered a risk rate which represents the risk probability of a task t_i of the lth security service. The **`computeRisk()`** method of the **`TaskSec`** class was implemented considering different security services computed by Eq. 3.

$$P(t_i, sl_i^l) = 1 - exp(-\lambda^l(sr_i^l - sl_i^l)), \quad l \in \{a, g, c\} \tag{3}$$

The risk coefficient λ^l can vary from one security service to another. For instance, a VM may be attacked three times by snooping attacks, two alteration attacks, and four spoofing attacks in a given period. The probability of a security failure is represented by negative exponent and it grows with the difference $sr_i^l - sl_i^l$, where sr_i^l is the security level required to a task t_i in each l security service, and sl_i^l represents the security level granted to a task t_i for each l security service. And, the risk probability of a task t_i is computed by Eq. 4.

$$P(t_i) = 1 - \prod_{l \in \{a,g,c\}} (1 - P(t_i, sl_i^l)) \tag{4}$$

To compute the overall risk of workflow execution, Eq. 5 is assumed.

$$P(T) = 1 - \prod_{t_i \in T} (1 - P(t_i)) \tag{5}$$

The methods of the **`WfExecStats`** class are called to show the execution summary. We run simulations using the Epigenomics workflow with 997 tasks with 1TB of data per task. We varied the number of required secure tasks for evaluating the overhead influence on makespan, cost, and risk of data leakage. We set the cheapest VM for all tasks and maximum level of security services for each secured required tasks. Figure 6(c) shows the risk of data leakage of a workflow execution assuming that all workflow tasks are sensitive. As can be seen, there is a high risk (about 100%) when less than 993 tasks are protected. The risk is only below 50% and closes to 0% when 995 and 997 tasks respectively are secured. This is typical behavior for the security metrics in computational systems, where any failure can result in significant losses for a user or organization. To evaluate the overhead produced by security services applied to tasks, we present the makespan analysis in Figure 6(a). As expected, the execution time grew as the number of tasks protected increased. The makespan was more prolonged when the latest workflow tasks were secured because they have the highest amount of output data

where the most costly security service (encryption) is applied. The total execution cost was also evaluated and shown in Figure 6(b). Because the virtual machine billing model is based on hours of use, it can be noted that the execution cost is approximately US$217 for 720 secured tasks, and subsequently increased significantly up to US$250 for all tasks secured. This was due to increased execution time by security services and consequently resulted in another hour of virtual machine rentals. It is important to note that the types of virtual machines could be optimized so that tasks requiring more processing were executed in faster instances. However, since the objective of this study is to evaluate the impact of security services on the simulations, we only set one type of instance for execution.

VII. Concluding Remarks

Workflow scheduling algorithms with awareness of security aspects have been investigated recently. Studies that involve security-based scheduling algorithms consider the application of security services typically to sensitive tasks. In this paper, we have proposed an extension for simulation of security overhead and risk analysis in workflow executions. It provides an extensible structure, which allows adding security services parameters and VM types for workflow simulation. Besides, the adaptability of this extension allows the implementation of different models of risk analysis. The risk model implemented shows that it is necessary to ensure almost all workflow tasks so that the security risk is reasonably reduced.

Although our proposed extension addresses security overheads by applying security services to sensitive tasks in a unique cloud provider, there is another approach for securing workflow execution, which is based on allocating sensitive tasks to private clouds and the non-sensitive ones to public clouds. Then, further research should be carried out to compute cost, makespan and security implications allocating tasks to hybrid clouds, bearing in mind that data transmission between tasks is also risky.

TABLE I
Sample list of security algorithms for workflow scheduling implemented in the WfSecurity extension.

Security Service	Algorithm	Security level	Throughput (kB/ms)
Encryption	SEAL	0.08	168.75
	RC4	0.14	96.43
	Blowfish	0.36	37.50
	Knufu/Khafre	0.40	33.75
	RC5	0.46	29.35
	Rijndael	0.64	21.09
	DES	0.90	15.00
	IDEA	1.00	13.50
Integrity verification	MD4	0.18	23.90
	MD5	0.26	17.09
	RIPEMD	0.36	12.00
	RIPEMD128	0.45	9.73
	SHA1	0.63	6.88
	RIPEMD160	0.77	5.69
	TIGER	1.00	4.36
Authentication	HMAC_MD5	0.55	90 (ms)
	HMAC_SHA_1	0.91	148 (ms)
	CBC_MAC_AES	1.00	163 (ms)

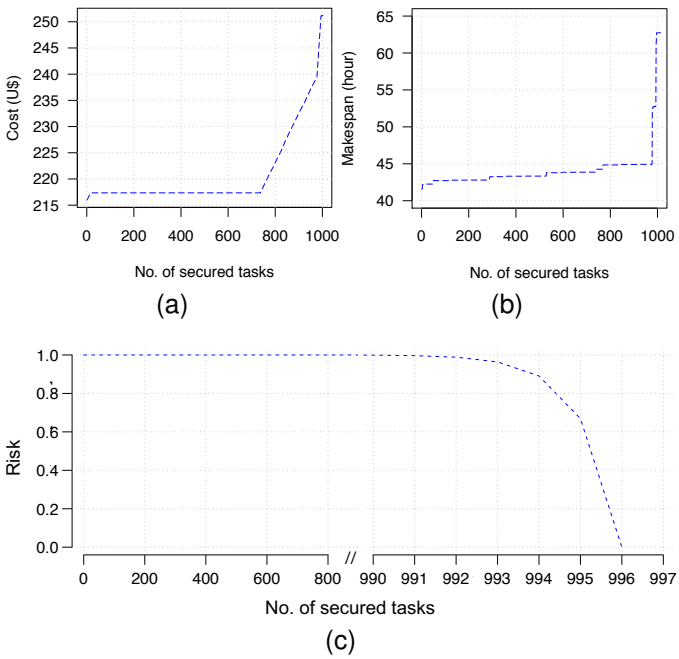

Fig. 6. Workflow execution metrics. (a) Monetary cost applying security services in workflow's tasks; (b) Makespan applying security services in workflow's tasks; (c) Risk applying security services in workflow's tasks.

Acknowledgments

The authors acknowledge CAPES, FAPESP (2011/09524-7 and 2013/01818-7), CEPID-CeMEAI (2013/07375-0), and CNPq for the resources provided, USP for the infrastructure offered, and UTFPR for the scholarship awarded to Henrique Yoshikazu Shishido.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] 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.

[3] L. Bertram, A. Ilkay, B. Chad, H. Dan, J. Efrat, J. Matthew, L. E. A., T. Jing, and Z. Yang, "Scientific workflow management and the kepler system," *Concurrency and Computation: Practice and Experience*, vol. 18, no. 10, pp. 1039–1065, 2006.

[4] 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.

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

[6] M. Rodriguez and R. Buyya, "A taxonomy and survey on scheduling algorithms for scientific workflows in iaas cloud computing environments," *Concurrency Computation*, vol. 29, no. 8, 2017.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] H. Y. Shishido, J. C. Estrella, C. F. M. Toledo, and M. S. Arantes, "Genetic-based algorithms applied to a workflow scheduling algorithm with security and deadline constraints in clouds," *Computers & Electrical Engineering*, vol. 69, pp. 378 – 394, 2018.

[14] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience*, vol. 41, no. 1, pp. 23–50, 2011.

[15] W. Chen and E. Deelman, "Workflowsim: A toolkit for simulating scientific workflows in distributed environments," in *IEEE 8th International Conference on E-Science*, 2012, pp. 1–8.

[16] M. Bux and U. Leser, "Dynamiccloudsim: Simulating heterogeneity in computational clouds," *Future Generation Computer Systems*, vol. 46, pp. 85 – 99, 2015.

[17] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. F. da Silva, M. Livny, and K. Wenger, "Pegasus, a workflow management system for science automation," *Future Generation Computer Systems*, vol. 46, pp. 17 – 35, 2015.

[18] H. Y. Shishido, J. C. Estrella, and C. F. M. Toledo, "Multi-objective optimization for workflow scheduling under task selection policies in clouds," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, 2018, pp. 1–8.

[19] H. Y. Shishido, J. C. Estrella, C. F. M. Toledo, and S. Reiff-Marganiec, "(wip) tasks selection policies for securing sensitive data on workflow scheduling in clouds," in *2018 IEEE International Conference on Services Computing (SCC)*, 2018, pp. 233–236.

[20] Amazon. (2018, jun) Instances types of amazon ec2. [Online]. Available: https://aws.amazon.com/pt/ec2/instance-types/