Simulation Modelling Practice and Theory 93 (2019) 119–132

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

MOWS: Multi-objective workflow scheduling in cloud computing based on heuristic algorithm

Farzaneh Abazari[a], Morteza Analoui[*,a], Hassan Takabi[b], Song Fu[b]

[a] School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
[b] Department of Computer Science and Engineering, University of North Texas, Denton, Texas, US

ARTICLE INFO

Keywords:
Cloud computing security
Secure task scheduling
Scientific workflows
Attack response

ABSTRACT

Cloud computing is emerging with growing popularity in workflow scheduling, especially for scientific workflow. Deploying data-intensive workflows in the cloud brings new factors to be considered during specification and scheduling. Failure to establish intermediate data security may cause information leakage or data alteration in the cloud environment. Existing scheduling algorithms for the cloud disregard the interaction among tasks and its effects on application security requirements. To address this issue, we design a new systematic method that considers both tasks security demands and interactions in secure tasks placement in the cloud. In order to respect security and performance, we formulate a model for task scheduling and propose a heuristic algorithm which is based on task's completion time and security requirements. In addition, we present a new attack response approach to reduce certain security threats in the cloud. To do so, we introduce task security sensitivity measurement to quantify tasks security requirements. We conduct extensive experiments to quantitatively evaluate the performance of our approach, using WorkflowSim, a well-known cloud simulation tool. Experimental results based on real-world workflows show that compared with existing algorithms, our proposed solution can improved the overall system security in terms of quality of security and security risk under a wide range of workload characteristics. Additionally, our results demonstrate that the proposed attack response algorithm can effectively reduce cloud environment threats.

1. Introduction

Cloud computing consists of distributed physical machines providing on-demand services, infrastructure, and a platform to users over a network. Infrastructure-as-a-Service (IaaS) in the cloud provides efficient and flexible computational resources to users. For example, Amazon Web Services and Google Compute Engine offer a full range of computing and storage services via virtual machine (VM) instances. Users can run large-scale workloads on virtual machines hosted by cloud infrastructure. Cloud infrastructure enables parallel processing of application tasks that improves application completion time [1]. IaaS users fit the needs and gain proper benefit of resource virtualization with reasonable cost. However, this has also brought many new challenges for workflow execution and performance optimization [2]. Information leakage and malicious alteration of sensitive data are two of the major obstacles in deploying applications in a distributed environment such as the cloud. Security and performance are the main goals of users in migrating to the cloud [3,4]. Therefore, several concerns in this area must be considered, such as secure intermediate data transfer

* Corresponding author.
E-mail addresses: f_abazari@iust.ac.ir (F. Abazari), analoui@iust.ac.ir (M. Analoui), hassan.takabi@unt.edu (H. Takabi), song.fu@unt.edu (S. Fu).

https://doi.org/10.1016/j.simpat.2018.10.004
Received 15 June 2018; Received in revised form 30 September 2018; Accepted 9 October 2018
Available online 10 October 2018
1569-190X/

and storage.

Cloud technology offers different methods in developing workflow, such as meta-heuristic, heuristic, and hybrid scheduling [5]. Workflow models are widely used for scientific, business, and engineering applications. Recently, the number of scientific workflows deployed in the cloud is rising [6,7]. Since cloud provides a shared infrastructure for its customers, establishing security for each application and its sensitive data must be considered during the scheduling process. In addition, transferring sensitive data among cloud components such as data centers with limited bandwidth is a security challenge. Another challenge is shared infrastructure in cloud, such that could threaten application security [8,9]. Most of the scheduling algorithms for the cloud infrastructure consider resource performance and disregard security concerns of workflow execution in cloud. Recently, more attention has been paid to security requirements in cloud scheduling [10].

Most studies in workflow scheduling have only focused on cost and makespan [11–14]. They usually propose a multi-objective scheduling scheme that reduces the cost while preserving quality of service (QoS). However, security is one of the QoS dimensions which they do not consider. Some of the recent work considers security in scheduling workflow in cloud infrastructure [10,15,16]. However, these studies did not investigate the effect of communication among tasks and VMs on the overall security of the cloud. Motivated by this challenge, in this paper we design a security model using the workflow graph property to measure the security sensitivity of tasks in the cloud. Scheduling techniques allow us to find the best place for tasks based on the security consideration and completion time. The impact of task interaction on preserving security is also studied in this paper.

VMs belonging to the same application that may locate on different or the same hosts communicate with each other. VMs establish a virtual network in order to transfer data between each one. Some reports indicate that host security mechanisms cannot monitor intra-VM communications, since traffic over virtual networks may not be visible to security protection mechanisms on the physical network [12,17–19].VM-level protection allows VMs and their running tasks to remain secure in shared and multi-tenant environments. It can address most of the attacks and make the development of host security services easier. Transferring data between tasks of an application is inevitable. If the security mechanisms of a VM are not strong enough, malicious tasks running on the VM or malicious users can alter or delete data for their malicious goal to take control of VMs by transferring infected files and data to them [20]. Intermediate data alteration or deletion by malicious tasks affect the flawless execution of related tasks. Related tasks are those which data is transferred between them. Affected tasks could be tasks co-located with the malicious task. Although this issue is important to cloud security, no previous papers have considered this. Therefore, in the first part of our model we aim to address VM and task interaction threats in workflow scheduling.

The primary protected elements in the cloud are VMs running on the physical machines in a data center. These VMs contain user's sensitive data. In order to secure VMs, different security mechanisms are provided. Due to limited resources and budgets in cloud, making all the VMs secure is impractical. To address this issue, there are different security mechanisms with different protection power in VMs as well as different computational resources and costs. Our first objective is to provide a new scheme to assign each task to a proper VM that minimizes the overall threat to cloud infrastructure, while maintaining a proper completion time. We aim to address VM and task interaction threats in workflow scheduling. The idea of this work is to assign a task to a part of the cloud so that its security requirement can be better met, meanwhile, completing promptly. We incorporate tasks communication threats in scheduling tasks on VMs. This measure reduces the application security risk and threat effectively.

The second part of our model focuses on responding to attacks. Strategies for attack response in computer networks can be formulated as optimization problems that apply to the rerouting of a network connection, placing VMs, immunizing, and installing anti-virus software. Currently, no scheduling algorithm exists that can support the cloud with attack response and threat analysis. When an unpredicted attack occurs in the cloud infrastructure, it is not cost-effective to stop all the tasks or migrate most of the tasks to more secure VMs. If the cloud provider does not implement a proper response method to the attack, this attack can cause further damage to the cloud. Response strategies can be formulated as a rescheduling problem. Our second goal is to develop an efficient response approach to cloud attacks caused by malicious tasks that reduce the threat in the cloud, without incurring too much overhead to cloud performance. To our knowledge, this is the first paper to address the workflow scheduling problem in response to malicious task in the cloud.

Our proposed security- and performance-aware scheduling model (MOWS) minimizes security threats in the cloud's heterogeneous environment while maintaining a reasonable workflow response time. In the scientific workflow scheduling, we need to consider the following questions: (1) how to assign tasks to VMs; (2) in what order the VMs should execute tasks considering the data dependencies among tasks; (3) what the attack response policy is. The main contributions of this work are:

- To present a security- and performance-aware scheduling algorithm that minimizes the overall security threat from related tasks based on the heterogeneous infrastructure of the cloud, while maintaining a proper makespan (maximum tasks completion time), and providing better security for applications in the cloud;
- To provide security for tasks based on three aspects: confidentiality, integrity and availability;
- To respond to malicious tasks by rescheduling tasks to reduce the effect of the attack; and
- To examine proposed schemes by performing experiments with real-world applications and showing that our approach can provide security and improve performance concurrently for tasks.

The rest of the paper is organized as follows. We explore the background and related work in Section 2. In Section 3, the system architecture and definition are introduced. Next, in Section 4 we propose MOWS and attack response strategy. Section 5 discusses the results, and finally, Section 6 concludes the paper.

2. Background and related work

A considerable amount of literature focuses on designing new methods of workflow scheduling under specific budget constraints [21,22]. Lin and Wu [23] investigated a new model for minimizing workflow end-to-end delay under user-specified financial constraints. Zheng et al. [24] presented another method in a dynamic resource price environments. Byun et al. [25] investigated a scheduling method to minimize the financial cost on the user's end and to maximize the resource utilization on the cloud provider's end. Fard et al. [26] introduced a multi-objective scheduling algorithm in grid and cloud that optimizes makespan, economic cost, energy consumption, and reliability. Many researchers now believe that rather than using a performance approach, it might be more critical to consider security as an important factor in scheduling [10,15,16].

Earlier work has shown that workflow is subjected to security attacks [27]. According to Zhu et al. [20], there are several attack scenarios that threaten the security of workflow. Malicious users can create or execute illegal tasks that may cause severe damage to the execution of workflow and the data among them. Malicious tasks may corrupt intermediate data and affect other tasks. Presence of data dependencies in workflow applications intensifies the security attack threat and the level of damage.

One of the first studies that considered security as a parameter in scheduling tasks in the distributed systems is presented by Xie and Qin [28]. The closest work to this paper was presented by Xiaoyong et al. [29], which developed an algorithm in distributed systems. They incorporated security awareness into task scheduling. The dynamic environment of the cloud makes it different from other distributed systems. In addition to this, our approach considers task interactions as a risk factor for tasks. Liu et al. presented a novel security constraint model and introduced several meta-heuristic adaptations to the particle swarm optimization algorithm to deal with the formulation of efficient schedules [30].

There have been several works on scheduling workflow in cloud infrastructure that consider security. To enhance the security of intermediate data, Liu et al. [15] presented a new data placement strategy in scientific cloud workflows that improved intermediate data security while ensuring the data transfer time among scientific workflows. *SABA* [10] presented a new workflow scheduling strategy that combined security and cost. Their method consisted of three phases. Firstly, a certain priority rank was calculated and assigned to each task. Secondly, each task was assigned to a VM that minimized the cost. In the last phase, tasks were rescheduled according to the dynamic demand. Chen et al. focused on data privacy protection in workflow. So they proposed a privacy and cost aware method based on genetic algorithm for data intensive workflow applications [31]. Chen et al. in their study [32], exploited idle time slots on resources, resulting from data dependencies among workflow tasks, to mitigate the impact of data encryption time on workflows' makespans. They devised a novel security-aware workflow scheduling algorithm including two phases: (1) task scheduling with selectively duplicating predecessor tasks to idle time slots; and (2) intermediate data encrypting by exploiting tasks' laxity time. *FFBAT* [33] proposed a security and cost aware scheduling algorithm for heterogeneous tasks in scientific workflow executed in a cloud. Their algorithm is based on the hybrid optimization approach, which combines Firefly and Bat algorithms. Xie et al. [34] introduced a novel dynamic security-aware scheduling algorithm, which was capable of achieving high security for real-time tasks while improving resource utilization. Watson [16] suggested a multi-level security model for scheduling tasks on federated clouds. Marcon et al. [35] presented workflow scheduling in a hybrid cloud. They considered dependencies among cloud components that bring new factors during specification, scheduling, and virtual machine provisioning. Sharif et al. offered an algorithm that preserves privacy in scheduling of workflows, while still considering customers' deadlines and cost [36]. Jianfang et al. [37] suggests scheduling algorithm of the cloud workflow using discrete particle swarm optimization, which achieves better performance in the security, completion time, cost, and load balancing. Shishido examines the effect of both Particle Swarm Optimization (PSO) and Genetic-based algorithms (GA) on attempts to optimize workflow scheduling [38]. Recently, Naidu et al. [39] proposed a modified PSO with scout adaptation algorithm, which uses a cyclic term called mutation operator, to schedule jobs in the cloud environment securely. A recent study by Wen et al. proposed a multi-objective privacy-aware workflow scheduling algorithm that provides cloud customers a set of Pareto trade-off solutions [40].

Some preliminary work was carried out recently on rapid response to attacks occurring in computer networks. Goldberg et al. [41] presented a new formulation to optimally respond to epidemics and cyber attacks. They proposed a decision maker to maximize network utility, while limiting the probabilities of nodes being infected. Later, Leyffer and Safro [42] developed a new practical response method to combat cyber attacks on weighted complex networks. Since there has been no detailed investigation about attack response methodology in cloud computing, we propose a response method when an attack occurs in the cloud while it is running a workflow.

So, we present the first security- and performance-aware scheduling method in a heterogeneous cloud environment that can reduce the overall security threat, risk, and also maintain a reasonable completion time.

3. System architecture: model and definitions

In this section, we first enumerate the threats of implementing application workflow on the cloud, and then, introduce the architecture of the security- and performance-aware scheduling model (MOWS). Finally, we present the scientific workflow model, system model, and definitions.

3.1. Threat model

Deploying a scientific workflow in the cloud could cause threats to data security [2,15]. Previous researches [43,44] have reported data security breach in the shared infrastructure of the cloud. Table 1 describes some security threats that may affect the

Table 1
Some security threats in workflow application.

Threat Type	Affected Service	Definition	Scenario
Information Disclosure	Confidentiality	unauthorized disclosure of information, including the user's authentication information and intermediate data among tasks, during task execution;	Malicious user or cloud insider successfully gain unauthorized access to the data in transmission between tasks. If the data is sensitive, disclosure of the data causes irreparable results.
Alteration	Integrity	modification of data to achieve a malicious goal;	Attacker or malicious task change the intermediate data over a network to affect the task that receives the data or it may infect the VM which is running the task.
Denial of Service	Availability	making data and resources unavailable to the tasks that need data to start; an attempt to make a resource unavailable to its intended task;	Attacker can create or execute illegal tasks that prevent relevant task from accessing intermediate data and may cause severe damage to the execution of the application and data.

proper execution of workflows on the cloud. Varadharajan and Tupakula in [19] believed that a malicious user who has obtained access to the privileged domain can perform different attacks on the co-located tenant virtual machines. Our scheduling model aims to reduce these threats while maintaining a reasonable response time for tasks. We suppose that an attacker can be a malicious user, or a cloud insider.

Sensitive tasks are those with sensitive input or output data. The output data for a task will be the input data for other tasks, so the information is traversing the network. The data, which will be used by more tasks in the future, should be generated, stored, and transmitted in a secure way. Information leakage or tampering in a data intensive application may cause damage to the application process. As mentioned before, the information is likely to be changed or leaked by malicious tasks. The tampered data can affect other tasks that use this data. Then, the threat spreads among more tasks of the application and compromises the application function and security. If there are any integrity, confidentiality, or authentication mechanisms among the VMs, the probability of these threats lessens.

As a matter of fact, data centers offer encrypted data storage, security management, and audit services to their customers. Recorded information about who has performed which actions on the data can be analyzed by the cloud provider to detect malicious behavior [45]. Since implementing security mechanisms causes overhead and cost in the cloud infrastructure, the level of security service is not the same for all VMs. The cloud provider knows the tasks' security demands and the relationship between the tasks, so it can predict which tasks are more vulnerable or sensitive to attacks. Cloud providers must consider this information in the assignment of each task to a proper VM.

3.2. MOWS architecture

In this paper, we consider an IaaS cloud system, which consists of several data centers with storage and computing resources that are presented by VMs. We propose a security-aware scheduling that investigates the effect of task interaction in the security risk of the cloud. In this section, we provide a high-level overview of MOWS. Fig. 1 illustrates the MOWS architecture, which consists of 5 components:

1. **Execution time manager (ETM)** is used to accept task information and calculate the complexity of each task. Task complexity directly affects the task execution time. It can be defined as a number of instructions in the task that should be executed by CPU.
2. **Communication time manager (CTM)** is responsible for evaluating task communication time with other tasks when transferring data among each other.
3. **Security overhead manager (SM)** is used to calculate task security risk in each VM.
4. **Attack response (AR)** is responsible for measuring the threat level of all tasks after an unusual behavior is detected in the cloud by a security administrator. The security mechanisms may be used to monitor the behavior of tasks and detect malicious entities.

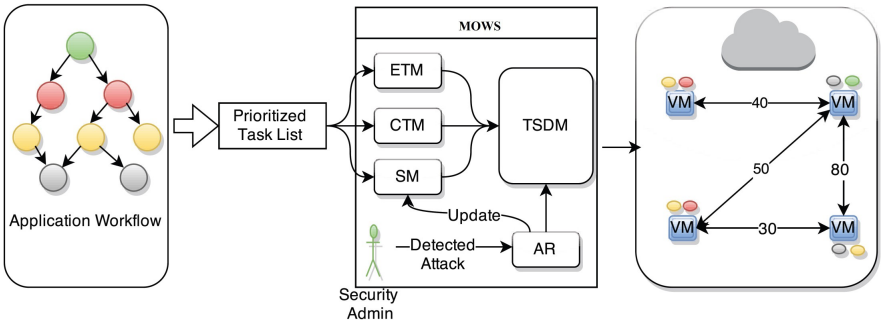


Fig. 1. MOWS architecture.

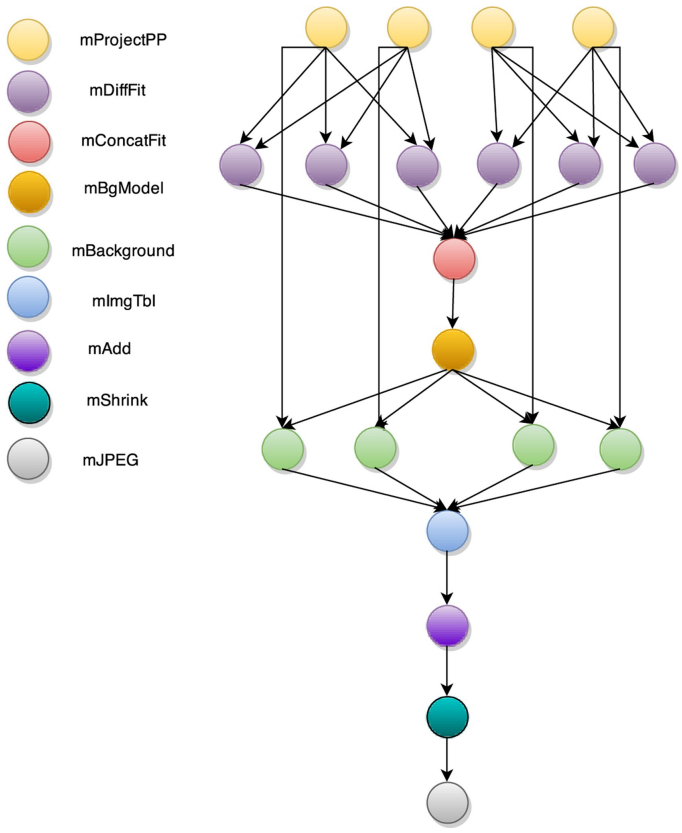


Fig. 2. Montage workflow.

The module should determine those tasks which have a high risk of being attacked. This module outputs high-risk tasks to the task scheduling decision maker module.

5. **Task scheduling decision maker (TSDM)** accepts inputs from other modules and VMs and assigns each task to a proper VM based on the information from the other modules to improve the security and performance of the cloud.

Although it is important to respond to attacks in a shared infrastructure environment such as a cloud, no previous architecture considered an attack response module. Without an attack response module, the security threat will increase in the cloud infrastructure after the initial attack is detected.

3.3. Scientific workflow model based on DAG

When a client submits his application to a cloud, it will be partitioned into several tasks. The relationship among tasks is determined by the application workflow. If tasks coordinate properly, the workflow will execute correctly. Tasks communicate with each other by transferring data [20]. Scientific workflows compose and execute a series of computational or data manipulation phases in a scientific application and can be modeled as a directed acyclic graph (DAG). We want to propose a solution that schedules scientific workflow as a data and computation-intensive workflow in the cloud. Fig. 2 depicts a scientific workflow DAG called Montage [46], which consists of nine types of tasks. These tasks cooperate with each other by sending and receiving data. Basic concepts about our system model are described in the following section.

3.4. System model and definition

Definition 1. A scientific application W_i is a set of tasks T, which are dependent upon each other. Application workflow can be modeled by DAG. The dependencies among tasks can be represented by the directed edges of the DAG .This connection reflects data flow *DS* through tasks. The child list and the parent list are determined for each task in DAG. The depth of the task is also determined by its location (distance to root) in DAG.

$$T = \{T_1, T_2, ..., T_m\} \ m\text{:number of tasks} \tag{1}$$

$$W_i = <T, DAG, DS> \tag{2}$$

The structure of scientific workflow is recognized by the number of inputs, the number of tasks, the data flow dependencies among tasks, and the execution time for each task. Task security requirements can be divided into several security services such as data integrity, confidentiality, and availability. Each service can be implemented by several algorithms with different overheads and strengths [47]. In order to incorporate security requirements into task specification, we assume that the security service demand (i.e., confidentiality, integrity and availability) for each task is specified as a set of values: low, medium, and high.

Definition 2. A scientific task consists of *IDS* and *ODS*, which are input and output data size, respectively. Each task has one or more parents, which send data to the task. In addition, tasks can have children send data to them. Task complexity represented by CompC is determined according to the execution time for each task. Users determine the security demand for each task based on the type of the task and the importance of the data that is generated by this task. Security demand *SD* represents the security service requirement of a task. Users can request different security services according to the sensitivity of a task. Confidentiality, Integrity and Availability are the three basic requirements. These requirements are specified from low to high by users and then normalized into the range [0,1]. High security demands increase the users' cost. This fact prevents users to request high security demand for all of the tasks.

$$T_i = <IDS_i, ODS_i, CompC, SD_i> \tag{3}$$

$$\begin{aligned} SD_i &= < SD^1, \ldots, SD^k > e.g. <Conf.\ ,Int.\ ,Avail.> \\ SD^k &: kthsecurityrequirement,\ k \in \{1, 2, 3\} \end{aligned} \tag{4}$$

Definition 3. Task specification is a tuple defined as:

$$TaskSpec_i = (CompC_i, CommC_i, SS_i), \tag{5}$$

where *CompC* represents task complexity, *CommC* represents task communication cost, and *SS* represents task security sensitivity. These components represent task characteristics that can be calculated using the following equations:

$$\begin{aligned} &CompC_i = Numberoftaskinstructions \\ &CommC_i = \sum IDS.\ Size \\ &SS_i = <SS_i^1, \ldots, SS_i^k > i\text{: } Tasknumber,\ k\text{: } securityrequirement,\ k \in \{1, 2, 3\} \\ &SS_i^k = \frac{SD^k \times (Max(Depth) - T_i.Depth) \times \#\, T_i.Parent()}{Max(Depth)} \end{aligned} \tag{6}$$

In order to change the value of *TaskSpec* elements between 0 and 1, all elements are divided by the maximum element among all tasks. The new vector is called $NormTaskSpec = (NCompC_i, NCommC_i, NSS_i)$, which has values between 0 and 1.

The security of resources can be measured with appropriate metrics [28]. Different security services provide different capabilities to prevent attacks. For example, if a VM uses SHA-1 for integrity service and another VM uses MD4, the first VM provides a higher level of security than the second VM. The security level of a VM is determined by a cloud provider based on the type of security services implemented in the VM.

Definition 4. VM specification is a tuple defined as:

$$VMSpec_i = (MIPS_i, BW_i, SeS_i), \tag{7}$$

where *MIPS* presents million instructions per seconds (to show the computing speed), *BW* indicates bandwidth, and *SeS* represents the level of security services provided by a cloud provider for virtualized infrastructure, which is a number between 0 and 1. The security level offered by a VM can be computed based on several capabilities, such as: intrusion detection system (IDS), antivirus, and firewall. However this is out of the scope of this paper and we assume that in each VM there is a mechanism to aggregate the security parameters of the VM and normalize it within the range [0, 1]. The greater value of *SeS* expresses the higher level of security services. In order to change the vector's elements to have a value between 0 and 1, all the elements are divided by the maximum element among all VMs. The new vector is called $NormVMSpec = (NMIPS_i, NBW_i, NSeS_i)$, which has values between 0 and 1.

3.5. Deficiency degree

In order to consider security in task scheduling, a model is needed to assess the amount of security that must be provided by the cloud provider for the task. Since a position and interaction of each task in the workflow is important in the task security sensitivity, we propose a model of task security based on the position of the task in DAG, which is introduced in Section 3.4. In addition to security concerns, tasks need different levels of computational power. In application workflow, there are different kinds of tasks with different levels of complexity. Tasks with high complexity have longer execution time in weak VMs. Consequently, other tasks that are dependent upon these tasks must wait longer and complete later. The cloud provider is aware of the complexity of the task and computation power of the VMs. Another characteristic of a scientific workflow is data intensity. Generated data in tasks should be transmitted to other tasks which are running on other VMs. This transmission occurs in the virtual network between VMs. Network bandwidth will affect the transmission time of intermediate data. To address these issues in scheduling tasks, two vectors (*VMSpec*) and (*TaskSpec*) were presented before. The first vector is related to task specification, and the second is related to VM resource capacities.

In order to schedule a task T_i in a proper VM_j, we propose a metric called Deficiency Degree (DD), which is calculated using

(*NormVMSpec*) and (*NormTaskSpec*). The *DD* is a weighted sum of Computation Deficiency (*CD*) (Eq. (8)), Transmission Deficiency (*TD*) (Eq. (9)), and Security Deficiency (*SD*) (Eq. (10)). In Eq. (11), W_{Per} and W_{Sec} are determined by the user. This equation shows the importance of security or performance from the user's point of view.

$$CD_{(i,j)} = \begin{cases} NMIPS_j - NCompC_i, & \text{if } NMIPS_j > NCompC_i \\ 2, & otherwise \end{cases} \tag{8}$$

$$TD_{(i,j)} = \begin{cases} NBW_j - NCommC_i, & \text{if } NBW_j > NCommC_i \\ 2, & otherwise \end{cases} \tag{9}$$

$$SD^k_{(i,j)} = \begin{cases} NSeS^k_j - NSS^k_i, & \text{if } NSeS^k_j > NSS^k_i \\ 2, & otherwise \end{cases} \tag{10}$$

$$\begin{aligned} DD_{T_i,VM_j} &= W_{Per} \times (CD_{(i,j)} + TD_{(i,j)}) \\ &\quad + W_{Sec} \times \textstyle\sum_{k=1}^{3} SD^k_{(i,j)} \end{aligned} \tag{11}$$

$$W_{Per} + W_{Sec} = 1 \tag{12}$$

Since a scientific workflow is a collection of dependent tasks, the total deficiency degree for workflow W_i called *DDW* is defined as:

$$DDW(W_i) = \sum_{j=1}^{m} DD(T_j) \tag{13}$$

For each task, a small *DD* value means a high satisfaction degree. Small *DD* value implies that a task's security and performance requirements can be perfectly met by the VM that hosts the task.

4. Proposed approach: MOWS

Data centers in cloud have different architectures, so VMs in these data centers have different resource capacity and security levels. Some VMs implement SSL to ensure data security, whereas other VMs use a virtual network that is implemented without any security services. We present a new method to utilize security services in those VMs, which are more secure, and schedule sensitive tasks on those VMs.

Eq. (14)shows the objective function of MOWS.

$$Minimize \quad DDW(W_i) \tag{14}$$

In other words, our objective function incorporates computation time and data transmission cost while guaranteeing the data security:

$$\begin{aligned} &Minimize \quad ComputationTime(), \\ &Minimize \quad TransmissionTime(), \\ &Minimize \quad SecurityRisk(). \end{aligned} \tag{15}$$

4.1. Problem description

Depending on the type of analytical modeling techniques used in research, various problem-solving solutions are applied to solve scheduling algorithms [48]. These solutions include greedy algorithms, linear programming, and evolutionary algorithms. We use list scheduling as a greedy algorithm to solve our scheduling algorithm. List scheduling consists of two phases: a task prioritization phase, wherein a certain priority is computed and assigned to each task in DAG; and the machine assignment phase, wherein each task (in order of priority) is assigned to a machine that minimizes the cost function (Eq. (14)). In this section, we focus first on task prioritization. After that, we describe the MOWS algorithm in detail.

4.2. Task prioritization

We consider the following assumptions in our scheduling method:

- Several tasks can be scheduled on the same VM; however, they run in a particular order.
- A task cannot start until all input data is received.
- A task cannot start until all predecessor tasks are executed.

Based on these basic assumptions, we assign a rank to each task by bottom-up traversal of the DAG. Unlike other scheduling methods, we add security to the task rank. In this case, if two tasks have the same complexity and communication cost, the one with

Input : *PrioritizedList* of Tasks based on Rank (Equation16),
VMList, and *VMS Pec*
Output : Allocation Matrix
for each task $T_i \in T$ **do**
Calculate CompC(T_i) based on Equation 6;
Calculate CommC(T_i) based on Equation 6;
Calculate SS(T_i) based on Equation 6;
end for
for each task $T_i \in PrioritizedList$ **do**
  for each VM $VM_j \in VMList$ **do**
    Use Equation 11 to calculate $DD(T_i, VM_j)$
  end for
  Select VM_k that has the minimum $DD(T_i, VM_j)$
  Schedule T_i on VM_j. ($Allocation[i][k] = 1$)
  Update $VMS\ pec_k.MIPS$ (Decrease it by factor of α)
end for

Algorithm 1. MOWS algorithm.

higher security demand gets higher rank, so it schedules sooner than the other one. As a result, the task with higher security demand will schedule on a more secure VM.

$$rank_u(t_i) = \bar{w}_i + \max_{t_j \in succ(t_i)} (\bar{c}_{i,j} + rank_u(t_j)) + \sum_k SD_i/k \tag{16}$$

where $succ(t_i)$ is the set of immediate successor tasks of task t_i, w_i is the average of task t_i execution time on all of the VMs, and $c_{i,j}$ refers to average communication time between t_i and t_j.

4.3. MOWS algorithm

In this section, we present MOWS algorithm, which incorporates the security and performance requirements into cloud scheduling. MOWS pseudo-code is given in the Algorithm 1. In this algorithm, each tasks in *PrioritizedList* is scheduled on the VM that has the minimum *DD*.

4.4. Attack response algorithm

Response to security threats in the cloud is an important issue for cloud providers. If the cloud provider cannot cease the malicious entity from unauthorized access to resources and data (which is a threat for the whole workflow), it may lead to further damage through malicious alteration or deletion of sensitive data. In this section, we suppose that a cloud administrator detects malicious behavior in a VM that is running task [49,50]. In order to respond to this misbehavior, we divide tasks into three groups: completed tasks, running tasks, and future tasks. There is no strategy for completed tasks. To reduce the threat that a malicious task could have for other tasks such as intentional data modification to achieve a malicious goal, or making the data unavailable to other tasks, we reschedule high risk tasks to other VMs. Rescheduling running tasks is done by task migration with overheads on network links of the data center as well as on the CPU cycles of servers executing the migration. As a consequence, we assign a probability of being affected to each task and migrate those running tasks with higher probability of being affected to a more secure VM. The future tasks with higher probability must be rescheduled to reduce the overall threat in the cloud. Accordingly, the security requirement of the task is updated to make changes in the scheduling parameters. Consequently, these tasks will be located in a more secure VM.

Algorithm 2shows our proposed attack response approach. In this algorithm high risk tasks are detected and reschedule.

4.5. Application security analysis

Since cloud provides computational resources to execute a broad spectrum of applications from different customers, application vulnerabilities and malicious users can threaten the security of sensitive data. Vulnerability in one of the tasks can be exploited by a malicious user. This may cause further damage to the cloud infrastructure and other tasks. In other words, an attacker gains access to intermediate data generated by a task and maliciously change data to affect related tasks and running VMs. Deploying several security services is critical for preventing malicious entities from unauthorized access to the sensitive data. To evaluate the security of our approach two metrics are defined. The first one is “Security Risk” which evaluates the application workflow risk probability. The second one is “Security Threat” that evaluates the average threat for all of the tasks in the workflow after one of them become malicious.

Input : Detected Malicious Task
RunningList and $FutureList$
Output : New Allocation Matrix
for each task $T_i \in T$ **do**
  Calculate $SecurityThreat(T_i)$;
  if $T_i.SecurityThreat > Threashold$ **then**
    $HighRiskList.add(T_i)$
  end if
end for
for each task $T_j \in RunningList \cap HighRiskList$ **do**
  $Update(TaskSpec(T_j).SS)$
  $Migrate(T_j)$ based on the new $TaskSpec$
  $Update(AllocationMatrix)$
end for
for each task $T_j \in FutureList \cap HighRiskList$ **do**
  $Update(TaskSpec(T_j).SS)$
  $Reschedule(T_j)$
  $Update(AllocationMatrix)$
end for

Algorithm 2. Attack response algorithm.

4.5.1. Application security risk probability

In order to measure the security risk of the application, we propose a risk probability formula to evaluate the risk of the scheduling task T_i on VM VM_j. Shameli and Cheriet in [51] proposed an incremental approach which enabled cloud providers to assess and manage cloud security risks. The first step in this method is to indicate the importance of the asset, vulnerability, and threat. Importance of the asset can be determined by users via the amount of security demand, but indicating vulnerability of VMs is complicated. So in order to analyze the security of the workflow, we adopt the same approach as Xiaoyong et al. [29] but we also consider the effect of tasks communication on the tasks scheduling.

We assume that if security demand SD_i^k for task T_i is less than the security service SS_j^k (which are offered by a VM_j), the risk probability of scheduling T_i on VM_j is zero. On the other hand, risk probability increases exponentially with the difference $SS_i^k - SD_i^k$. The risk probability of scheduling T_i on VM_j is calculated using Eq. (18).

$$Pr^k(T_i^k, VM_j^k) = \begin{cases} 0, & if SD_i^k \le SS_j^k \\ 1 - e^{-(SD_i^k - SS_j^k)} & otherwise \end{cases} \tag{17}$$

The task risk probability is a joint probability of all task risk probability for each security service. $Pr(T_i, VM_j)$ indicates the probability that task T_i will be attacked during its execution.

$$Pr(T_i, VM_j) = 1 - \prod_k (1 - Pr^k(T_i^k, VM_j^k)) \tag{18}$$

The application workflow risk probability is an average of all tasks risk probabilities. $Pr(W)$ determines the average probability of composed tasks being attacked in a workflow.

$$Pr(W) = \frac{\sum_{T_i \in T} Pr(T_i, VM_j)}{m} \tag{19}$$

4.5.2. Application security threat

The application security threat is another measurement to evaluate the security preserving of the scheduling algorithm. In order to calculate the security threat, we implement the following scenario.

- Consider one of the tasks malicious.
- Simulate the attack (alter or delete sensitive data) in the cloud and consider VM security services in controlling the attack and preventing the spread of malicious behavior among tasks.
- Calculate the number of affected tasks.
- Run the simulation several times. Average the results.

We suppose that, if the overall security demand of a task is higher than the overall VM security services, the task can be affected by accessing altered or deleted intermediate data (that is generated by malicious parent). Since the performance of each task is totally

related to its input data (that is generated by the parent task), the malicious behavior of the parent directly affects on the task trend. So, the threat spreads among tasks. The total number of affected tasks is calculated and is called AS security threat.

In Section 5, these two metrics are used to evaluate the security of MOWS.

5. Performance evaluation

In this section, we conduct a set of experiments using real world workflows to analyze MOWS. We evaluate the performance in terms of the makespan, security risk (see Eq. (19)), and security threat (see Section 4.5.2) by comparing with well-known algorithms HEFT [52] and security-aware HEFT (SAHEFT). HEFT's nature is non-security aware, it selects a virtual machine for a task without considering the task's security demands. HEFT chooses a VM that has the minimum completion time for the given task. In order to make a fair comparison, we modify the HEFT algorithm by adding security. If a task has the same completion time in two VMs, SAHEFT will schedule a task on a VM which provides more security services. So SAHEFT provide security in addition to performance in workflow scheduling. We use WorkflowSim [53] to implement our scheduling algorithm and compare the results.

In this paper, the experiments need to be repeatable in order to compare the algorithm with different parameters. In addition, implementing experimental environments is expensive and hard to conduct as resource conditions vary from time to time due to the cloud shared infrastructure. Hence, simulation tools play an important role in cloud area research. WorkflowSim is an extension of the CloudSim [54], cloud simulation toolkit, which simulates workflow management and scheduling in dynamic cloud environment. It supports DAG and simulates scientific workflows in distributed environments with better accuracy and wider support than existing solutions. WorkflowSim is promising in providing an evaluation platform for research areas such as scheduling algorithms and overhead robustness studies. A series of important planning and scheduling algorithms, such as HEFT, DHEFT, Min-Min, and Max-Min are implemented by WorkflowSim.

5.1. Data sets

We perform experiments with five real world scientific workflow applications in the Pegasus project:

- **CyberShake** [55]: used by the Southern California Earthquake Center to classify earthquake alarms.
- **Epigenomics** [56]: used DNA sequence lanes to generate multiple lanes of DNA sequences.
- **Montage** [46]: created by NASA/IPAC stitches to gather multiple input images to create custom mosaics of the sky.
- **Inspiral** [57]: used to generate and analyze gravitational waveforms from data collected during the coalescing of compact binary systems.
- **Sipht** [58]: used in bioinformatics to search for small untranslated bacterial regulatory RNAs.

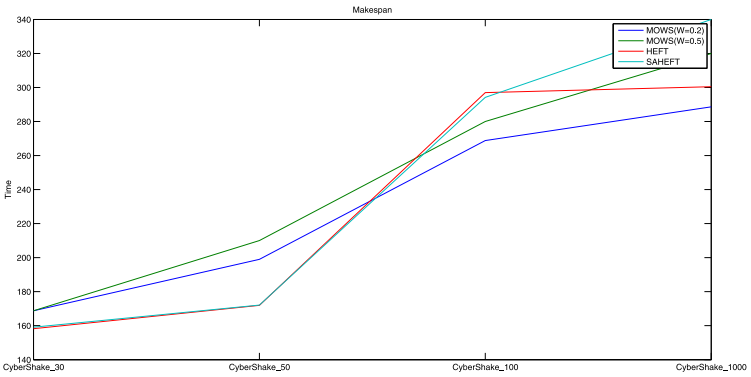
5.2. Experimental results

In order to show the functionality of the method, CyberShake workflows with 30, 50, 100, and 1000 tasks are applied. MOWS with $W_{Sec} = 0.2$ and $W_{Sec} = 0.5$ (W_{Sec} determines the effect of security in the method which is defined in Eq. (12)), HEFT, and SAHEFT are chosen to schedule these workloads on a heterogeneous cloud infrastructure with 20 physical and 100 virtual machines. To simulate the heterogeneity of the resources in the cloud, 9 different types of virtual machines in terms of CPU (MIPS), RAM, BW, and security services are chosen.

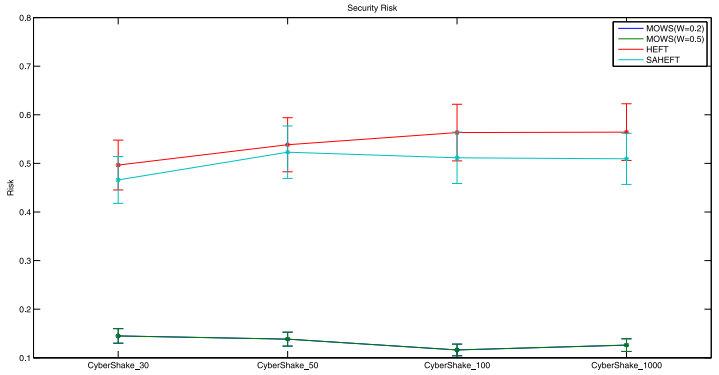
Results in Fig. 3 show the performance of the algorithms in terms of makespan, security threat, and security risk. Fig. 3a indicates that makespan of workloads in all of the algorithms are almost the same. The comparison among algorithms based on the security risk is illustrated in Fig. 3b. In MOWS the average probability of CyberShake workflow being attacked during its execution is less than 0.15, however this probability is much higher in HEFT and SAHEFT. Fig. 3c highlights that MOWS with $W_{Sec} = 0.5$ provides less security threat for all of the workloads. For example, security threat for CyberShake_1000 is less than one, that means the average threat for all of the tasks in the workflow after one of them become malicious is less than one task. However, scheduling with SAHEFT and HEFT has higher threat for workflow and more than 5 tasks are threatened after one of the tasks become malicious. Thus, our strategy can guarantee better security risk and threat than the other two strategies.

Fig. 4compares the performance of proposed MOWS with HEFT and the modified HEFT(SAHEFT). We apply MOWS ($W_{Sec} = 0.5$), HEFT, and SAHEFT to the DAG files of five real world workflows and calculate the makespan, security threat, and security risk. Each configuration was simulated 10 times to measure confidence interval. Fig. 4a presents the makespan of the three algorithms for different workflows on a cloud infrastructure with 50 virtual machines and 10 physical machines (considering a 95% confidence interval). As depicted, MOWS has a better makespan than the other two algorithms in Inspiral and Epigenomics workflows. In the other workflows they have almost the same makespan. Fig. 4b shows security risk in different workflows. Depending on the type of the workflows and the relationship between tasks, the security risk is different. MOWS has the best security risks among the algorithms (considering a 95% confidence interval). The results in Fig. 4c illustrate that the security threat in all of the workflows are minimum when the scheduling algorithm is MOWS (considering a 95% confidence interval). This improvement is due to the fact that MOWS approach is capable of employing the security attribute to expand the quality of scheduling. Since HEFT's nature is non-security aware, it selects a virtual machine for a task without considering the task's security demands. Always there is a trade-off between security and performance. We reduce the security threat and risk while maintaining a proper makespan for tasks.

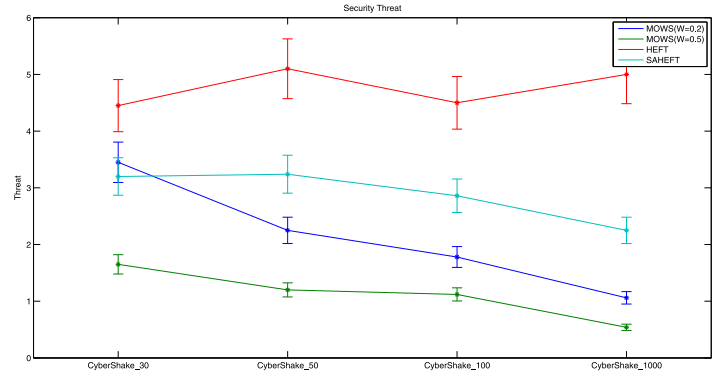
For evaluating the efficiency and overhead of our MOWS attack response algorithm, according to Section 4.4, we define a scenario



(a) Makespan



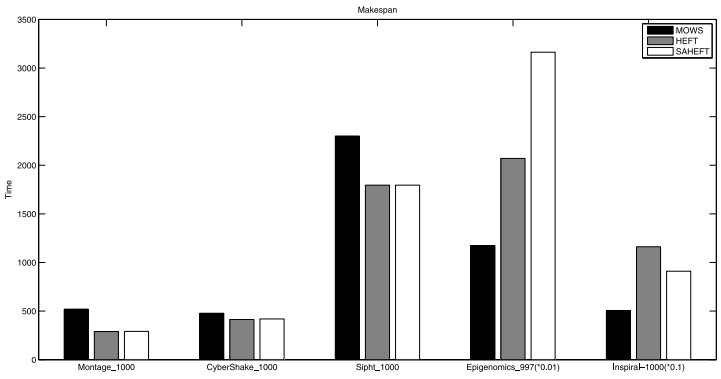
(b) Security Risk



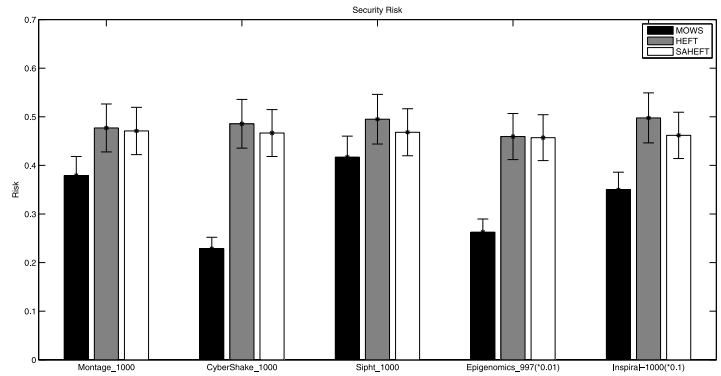
(c) Security Threat

Fig. 3. Simulation results on CyberShake.

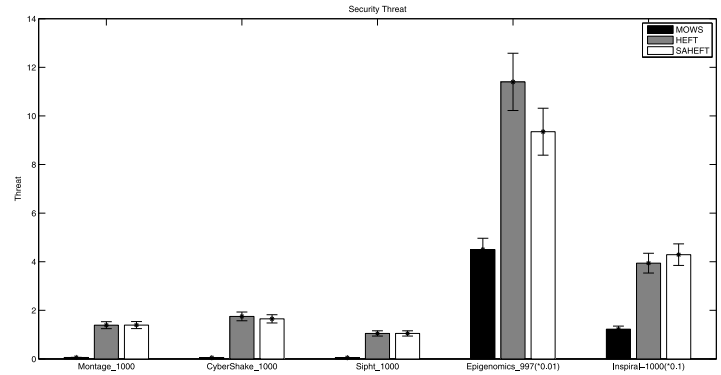
in the cloud environment where it is running Montage workflow (Fig. 2) with 1000 tasks. We assume one of the tasks is detected as a malicious task. The detected malicious task type is mDiffFit. This task may get malicious files from its parents. The attack response algorithm finds tasks with high-risk of being malicious via the detected task. Rescheduling high-risk tasks to a more secure VM helps the cloud environment to have less overall threat. We run a scenario with two conditions: (1) with attack response and (2) without attack response. Table 2 shows that security threat is reduced after the attack response; whereas, makespan increases (10%). As we mentioned earlier, after the attack is detected in the cloud, high risk tasks migrate to a more secure VM. Since these tasks restart in a



(a) Makespan



(b) Security Risk



(c) Security Threat

Fig. 4. Comparison of MOWS, HEFT, and SAHEFT .

Table 2
Attack response algorithm results (detected task with depth = 2).

	With attack response	Without attack response
Makespan	405	365
Security threat	44	110

Table 3
Attack response algorithm results (detected task with depth = 5).

	With attack response	Without attack response
Makespan	430	365
Security threat	35	72

new VM, they complete later. Table 3 represents the same scenario when the detected task has depth equal to five. The depth of the task is determined by its location. A task with smaller depth can affect more tasks in the application. Results in Table 3 confirm that a task with lower depth causes a higher security threat to the application. According to this table, the attack response module reduces the security threat from 72 to 35. Hence, using the attack response module can effectively reduce the security threat in application.

5.3. Complexity analysis

Theorem 1. *The time complexity of MOWS is O(m*n), which sets the number of tasks as m and the number of VMs as n.*

Proof. The time complexity of computing task's priority and sorting them based on their rank is $O(m + m \log m)$ (Steps 1). To compute task's CompC, CommC, and SS, the time complexity is $O(3*m)$ (Steps 5,6,7). Furthermore, Selecting VM for each task has time complexity $O(n)$. So, for all tasks, the time complexity is $O(m*n)$ (Steps 11,13). Thus, the total complexity of MOWS is $O(m*n)$. □

6. Conclusion

In a virtualized cloud environment, transferring data between tasks of an application is inevitable. If the security mechanism of a VM is not strong enough, the malicious task can affect other tasks by modifying intermediate data. VM-level protection allows VMs and tasks to stay secure in shared and multi-tenant environments. Although this issue is important to cloud security, none of the previous works has addressed it. We considered task interaction issues as a security threat, in addition to the completion time in workflow scheduling and presented MOWS, a security- and performance-aware scheduling method in a heterogeneous cloud environment that can reduce the overall security threat and risk. We designed a new systematic method that considers both tasks security demands and interactions in secure tasks placement in the cloud. To do so, we introduced task security sensitivity measurement to quantify tasks security requirements. The experimental results show that our algorithm can effectively reduce the security risk and threat compared to HEFT and security-aware HEFT, while maintaining a reasonable completion time. This study was limited by the absence of dynamic scheduling. An interesting future direction is to extend the MOWS algorithm for dynamic workflow scheduling that takes into account prediction in workflow behavior in the cloud environment. Moreover, a further study could evaluate the cost of our approach in pay-per-use clouds such as Amazon EC2 and Microsoft Azure.

References

[1] C. Chen, J. Liu, Y. Wen, J. Chen, Research on workflow scheduling algorithms in the cloud, Process-Aware Systems, Springer, 2015, pp. 35–48.
[2] Y. Zhao, X. Fei, I. Raicu, S. Lu, Opportunities and challenges in running scientific workflows on the cloud, Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on, IEEE, 2011, pp. 455–462.
[3] E. Caron, J.R. Cornabas, Improving users' isolation in IAAS: virtual machine placement with security constraints, Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on, IEEE, 2014, pp. 64–71.
[4] A. Singh, K. Chatterjee, Cloud security issues and challenges: a survey, J. Netw. Comput. Appl. 79 (2017) 88–115.
[5] M. Masdari, S. ValiKardan, Z. Shahi, S.I. Azar, Towards workflow scheduling in cloud computing: a comprehensive analysis, J. Netw. Comput. Appl. 66 (2016) 64–82.
[6] Q. Jiang, Y.C. Lee, M. Arenaz, L.M. Leslie, A.Y. Zomaya, Optimizing scientific workflows in the cloud: a montage example, Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on, IEEE, 2014, pp. 517–522.
[7] C. Szabo, Q.Z. Sheng, T. Kroeger, Y. Zhang, J. Yu, Science in the cloud: allocation and execution of data-intensive scientific workflows, J. Grid Comput. 12 (2) (2014) 245–264.
[8] E.M. Mohamed, H.S. Abdelkader, S. El-Etriby, Enhanced data security model for cloud computing, Informatics and Systems (INFOS), 2012 8th International Conference on, IEEE, 2012. CC–12
[9] S.K. Sood, A combined approach to ensure data security in cloud computing, J. Netw. Comput. Appl. 35 (6) (2012) 1831–1838.
[10] L. Zeng, B. Veeravalli, X. Li, Saba: a security-aware and budget-aware workflow scheduling strategy in clouds, J. Parallel Distrib. Comput. 75 (2015) 141–151.
[11] F. Zhang, J. Cao, K. Li, S.U. Khan, K. Hwang, Multi-objective scheduling of many tasks in cloud platforms, Future Gener. Comput. Syst. 37 (2014) 309–320.
[12] S. Yassa, R. Chelouah, H. Kadima, B. Granado, Multi-objective approach for energy-aware workflow scheduling in cloud computing environments, Sci. World J. (2013).
[13] L. Guo, S. Zhao, S. Shen, C. Jiang, Task scheduling optimization in cloud computing based on heuristic algorithm, J. Netw. 7 (3) (2012) 547–553.
[14] J. Liu, X.G. Luo, X.M. Zhang, F. Zhang, B.N. Li, Job scheduling model for cloud computing based on multi-objective genetic algorithm, IJCSI 10 (1) (2013) 134–139.
[15] W. Liu, S. Peng, W. Du, W. Wang, G.S. Zeng, Security-aware intermediate data placement strategy in scientific cloud workflows, Knowl. Inf. Syst. 41 (2) (2014) 423–447.
[16] P. Watson, A multi-level security model for partitioning workflows over federated clouds, J. Cloud Comput. 1 (1) (2012) 1–15.
[17] C. Wueest, Threats to Virtual Environments, (2014). http://www.symantec.com
[18] F. Abazari, M. Analoui, H. Takabi, Effect of anti-malware software on infectious nodes in cloud environment, Comput. Secur., 2016.
[19] V. Varadharajan, U. Tupakula, Security as a service model for cloud environment, IEEE Trans. Netw. Serv. Manage. 11 (1) (2014) 60–75.
[20] Y. Zhu, T. Xin, I. Ray, Recovering from malicious attacks in workflow systems, Database and Expert Systems Applications, Springer, 2005, pp. 14–23.

[21] L. Zeng, B. Veeravalli, X. Li, Scalestar: budget conscious scheduling precedence-constrained many-task workflow applications in cloud, Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on, IEEE, 2012, pp. 534–541.
[22] Y. Wang, W. Shi, On scheduling algorithms for mapreduce jobs in heterogeneous clouds with budget constraints, Principles of Distributed Systems, Springer, 2013, pp. 251–265.
[23] X. Lin, C.Q. Wu, On scientific workflow scheduling in clouds under budget constraint, Parallel Processing (ICPP), 2013 42nd International Conference on, IEEE, 2013, pp. 90–99.
[24] M. Zheng, J. CAO, Y. YAO, Cloud workflow scheduling algorithm oriented to dynamic price changes, Comput. Integr. Manuf. Syst. 8 (2013) 015.
[25] E.K. Byun, Y.S. Kee, J.S. Kim, S. Maeng, Cost optimized provisioning of elastic resources for application workflows, Future Gener. Comput. Syst. 27 (8) (2011) 1011–1026.
[26] H.M. Fard, R. Prodan, J.J.D. Barrionuevo, T. Fahringer, A multi-objective approach for workflow scheduling in heterogeneous environments, Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), IEEE Computer Society, 2012, pp. 300–309.
[27] M.S. Olivier, R.P. van de Riet, E. Gudes, Specifying application-level security in workflow systems, Database and Expert Systems Applications, 1998. Proceedings. Ninth International Workshop on, IEEE, 1998, pp. 346–351.
[28] T. Xie, X. Qin, Performance evaluation of a new scheduling algorithm for distributed systems with security heterogeneity, J. Parallel Distrib. Comput. 67 (10) (2007) 1067–1081.
[29] T. Xiaoyong, K. Li, Z. Zeng, B. Veeravalli, A novel security-driven scheduling algorithm for precedence-constrained tasks in heterogeneous distributed systems, computers, IEEE Trans. 60 (7) (2011) 1017–1029.
[30] H. Liu, A. Abraham, V. Snášel, S. McLoone, Swarm scheduling approaches for work-flow applications with security constraints in distributed data-intensive computing environments, Inf. Sci. 192 (2012) 228–243.
[31] C. Chen, J. Liu, Y. Wen, J. Chen, D. Zhou, A hybrid genetic algorithm for privacy and cost aware scheduling of data intensive workflow in cloud, International Conference on Algorithms and Architectures for Parallel Processing, Springer, 2015, pp. 578–591.
[32] H. Chen, X. Zhu, D. Qiu, L. Liu, Z. Du, Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds, IEEE Trans. Parallel Distrib. Syst. 28 (9) (2017) 2674–2688.
[33] A. Arunarani, D. Manjula, V. Sugumaran, Ffbat: a security and cost-aware workflow scheduling approach combining firefly and bat algorithms, Concurr. Comput. 29 (24) (2017) E4295.
[34] T. Xie, A. Sung, X. Qin, Dynamic task scheduling with security awareness in real-time systems, Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International, IEEE, 2005, p. 8.
[35] D.S. Marcon, L.F. Bittencourt, R. Dantas, M.C. Neves, E.R. Madeira, S. Fernandes, C.A. Kamienski, M.P. Barcelos, L.P. Gaspary, N.L.d. Fonseca, Workflow specification and scheduling with security constraints in hybrid clouds, Cloud Computing and Communications (LatinCloud), 2nd IEEE Latin American Conference on, IEEE, 2013, pp. 29–34.
[36] S. Sharif, J. Taheri, A.Y. Zomaya, S. Nepal, Mphc: preserving privacy for workflow execution in hybrid clouds, Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2013 International Conference on, IEEE, 2013, pp. 272–280.
[37] C. Jianfang, C. Junjie, Z. Qingshan, An optimized scheduling algorithm on a cloud workflow using a discrete particle swarm, Cybern. Inf. Technol. 14 (1) (2014) 25–39.
[38] H.Y. Shishido, J.C. Estrella, C.F.M. Toledo, M.S. Arantes, Genetic-based algorithms applied to a workflow scheduling algorithm with security and deadline constraints in clouds, Comput. Electr. Eng. (2017).
[39] P.S. Naidu, B. Bhagat, Secure workflow scheduling in cloud environment using modified particle swarm optimization with scout adaptation, Int. J. Model. Simul. Sci. Comput. 9 (01) (2018) 1750064.
[40] Y. Wen, J. Liu, W. Dou, X. Xu, B. Cao, J. Chen, Scheduling workflows with privacy protection constraints for big data applications on cloud, Future Gener. Comput. Syst. (2018).
[41] N. Goldberg, S. Leyffer, I. Safro, Optimal Response to Epidemics and Cyber Attacks in Networks, preprint ANL/MCS-1992-0112, Argonne National Laboratory, Mathematics and Computer Science Division, 2012.
[42] S. Leyffer, I. Safro, Fast response to infection spread and cyber attacks on large-scale networks, J. Complex Netw. 1 (2) (2013) 183–199.
[43] M. Ali, S.U. Khan, A.V. Vasilakos, Security in cloud computing: opportunities and challenges, Inf. Sci. 305 (2015) 357–383.
[44] F. Lombardi, R.D. Pietro, Secure virtualization for cloud computing, J. Netw. Comput. Appl. 34 (4) (2011) 1113–1122.
[45] R.A. Popa, J.R. Lorch, D. Molnar, H.J. Wang, L. Zhuang, Enabling security in cloud storage slas with cloudproof. USENIX Annual Technical Conference, Vol. 242, (2011), pp. 355–368.
[46] J.C. Jacob, D.S. Katz, T. Prince, G.B. Berriman, J.C. Good, A.C. Laity, E. Deelman, G. Singh, M.H. Su, The Montage Architecture for Grid-Enabled Science Processing of Large, Distributed Datasets, Pasadena, Jet Propulsion Laboratory, National Aeronautics and Space Administration, CA, 2004.
[47] M. Qiu, L. Zhang, Z. Ming, Z. Chen, X. Qin, L.T. Yang, Security-aware optimization for ubiquitous computing systems with seat graph approach, J. Comput. Syst. Sci. 79 (5) (2013) 518–529.
[48] M.H. Ferdaus, M. Murshed, Energy-aware virtual machine consolidation in IAAS cloud computing, Cloud Computing, Springer, 2014, pp. 179–208.
[49] M. Bazm, R. Khatoun, Y. Begriche, L. Khoukhi, X. Chen, A. Serhrouchni, Malicious virtual machines detection through a clustering approach, Cloud Technologies and Applications (CloudTech), 2015 International Conference on, IEEE, 2015, pp. 1–8.
[50] M.T. Khorshed, A.S. Ali, S.A. Wasimi, A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing, Future Gener. Comput. Syst. 28 (6) (2012) 833–851.
[51] A.S. Sendi, M. Cheriet, Cloud computing: a risk assessment model, Cloud Engineering (IC2E), 2014 IEEE International Conference on, IEEE, 2014, pp. 147–152.
[52] H. Topcuoglu, S. Hariri, M.y. Wu, Performance-effective and low-complexity task scheduling for heterogeneous computing, Parallel Distrib. Syst. IEEE Trans. 13 (3) (2002) 260–274.
[53] W. Chen, E. Deelman, Workflowsim: a toolkit for simulating scientific workflows in distributed environments, E-Science (e-Science), 2012 IEEE 8th International Conference on, IEEE, (2012), pp. 1–8. Download from: http://www.github.com/WorkflowSim/
[54] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. De Rose, R. Buyya, Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software 41 (1) (2011) 23–50.
[55] H. Magistrale, S. Day, R.W. Clayton, R. Graves, The scec southern california reference three-dimensional seismic velocity model version 2, Bull. Seismol. Soc. Am. 90 (6B) (2000) S65–S76.
[56] Usc Epigenome Center, http://epigenome.usc.edu, accessed, (2018).
[57] D.A. Brown, P.R. Brady, A. Dietz, J. Cao, B. Johnson, J. McNabb, A case study on the use of workflow technologies for scientific analysis: gravitational wave data analysis, Workflows for e-Science, Springer, 2007, pp. 39–59.
[58] J. Livny, H. Teonadi, M. Livny, M.K. Waldor, High-throughput, kingdom-wide prediction and annotation of bacterial non-coding rnas, PLoS ONE 3 (9) (2008) E3197.