Received May 8, 2018, accepted June 11, 2018, date of publication June 21, 2018, date of current version July 12, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2849594

A Qos-Driven Approach to the Cloud Service Addressing Attributes of Security

HAN XU, XIWEI QIU, YONGPAN SHENG, LIANG LUO, AND YANPING XIANG
School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

Corresponding authors: Han Xu (xuhan200328@outlook.com) and Yanping Xiang (yanping_xiang@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61602094.

ABSTRACT Recently, cloud computing has been widely used by relying on its powerful resource integration and computing abilities. In the cloud computing system (CCS), the quality of service (QoS) is an important service evaluation criterion from provider and client perspectives, which directly affects the client experience and profit of the cloud providers. Thus, a precise evaluation of the QoS can help the cloud provider develop reasonable resource allocation strategies for improving the client experience. The performance metric is usually adopted to quantify QoS. Many approaches and methods for evaluating performance have been widely studied. However, another important metric, i.e., security, does not receive adequate attention in the evaluation of QoS. More importantly, security also has serious effects on the performance metric, that is, complex security-performance (S-P) correlations. To address these issues, this paper first builds a Markov model to analyze and assess the security of the CCS that captures two critical security factors, i.e., malicious attacks and the security protection mechanism. Then, a hierarchical modeling approach is presented to flexibly build the connection between security and the service performance. Finally, we propose a correlation metric to quantify random service performance. This correlation metric comprehensively considers the effect of the security factors and thus becomes more realistic and precise. The experimental results reveal the dynamic change of performance caused by the security factors and demonstrate the important S-P correlation. Therefore, security cannot be ignored in the modeling and evaluation of the QoS metric.

INDEX TERMS Cloud service, quality of service, security modeling, performance modeling.

I. INTRODUCTION

The CCS dynamically organizes VMs and servers to run a series of services, in which each service is separated to a set of tasks that are executed on VMs in parallel. Although dynamic organization and parallel execution makes that the CCS complete services effectively, it also brings a serious problem to CCS security. Due to virtualization of the CCS, VM security becomes a new security factor that is not present in the traditional system. VM security has some identical features, for example, different VMs are totally isolated and co-located VMs may fail simultaneously due to an attack on their host server. Meanwhile, VM security has been paid much attention in recent studies. Gonzales *et al.* [1] indicated that, on the IaaS level, most cloud specific attacks threaten CCS through compromised VM, such as VM side channel attack, VM attack through the hypervisor, disk injection to live VM, and many other attacks. VM side channel attack is usually based on VM vulnerabilities [2]. It is representative of a class of attacks that take advantage of VM co-residency, which arises when VMs of two or more users share the same hardware. VM attack through the hypervisor (HV) starts from the compromised VM [3], [4]. The attacker obtains valid government user credentials (through spearfishing, surveillance, or use of malware) from this compromised VM. Then, the attacker can obtain a public cloud account to compromise the HV and access the target co-residency agency VM from a related attack path present in the public cloud. In disk injection to live VM attack, the attacker attempts to gain access to target data by placing malicious code in the local attached storage of the targeted VM [5]. It is obvious that if attackers are willing, their actions will have a huge effect on the QoS of CCS through those compromised VMs. It is found that VM security seriously affects the security of the CCS. Therefore, how to assess the CCS security and evaluate its impact on the QoS metric, such as the service performance, are critical issues that must be solved to ensure the application of cloud computing in the real-world environment [6].

In recent years, the performance metric that is adopted to quantify QoS has been well studied. Hwang*et al.* [7] present generic cloud performance models for evaluating infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS), and mashup or hybrid clouds. They test clouds with real life benchmark programs and propose new performance metrics. Cao *et al.* [8] study the problem of optimal multiserver configurations for profit maximization in a cloud computing environment. In this approach, they build a pricing model and treat a multiserver system as an M/M/m queuing model to formulate and analytically solve the optimization problem. Hwang *et al.* [9] develop new performance models and run the HiBench benchmark to test Hadoop performance on various elastic compute cloud (EC2) configurations and evaluate three scaling strategies to upgrade the performance, efficiency and productivity of elastic clouds like EC2. In this approach, the construction of performance models are same as in [8]. In [10], Iosup *et al.* analyze the performance of cloud computing services for scientific computing workloads. They quantify the presence in real scientific computing workloads of many-task computing (MTC) users. Then, an empirical evaluation of the performance of four commercial cloud computing services including Amazon EC2 is performed. However, the impact of the security on the performance is not considered in these approaches.

Security requirements are often in contrast with performance requirements [11]. On the one hand, the operation of security mechanism inevitably consume certain amount of central processing unit (CPU) time and memory, and thus increase the service delay. On the other hand, malicious attacks may make servers or VMs be failed, and eventually lead to the service failures or overtime failures. Current researches on system security often separately consider the effect of security mechanisms and malicious attacks. Batista *et al.* [12] propose a Qos-driven approach to study the overhead imposed by security mechanisms on an performance of different systems. But, the authors have not studied the impact of malicious attacks on performance, which will results in an inaccuracy for the performance evaluation. Wolter *et al.* [13] set up a simple generalized stochastic petri net to illustrate general aspects of the security-performance tradeoff, in which the security model is used to evaluate the specific security mechanism. Yau *et al.* [14], Zeng *et al.* [15], and Cho *et al.* [16] take the encryption and decryption security mechanism into consideration and present an adaptive security-performance tradeoff model in different systems, i.e., service-based systems, distributed networked control systems, and mobile ad hoc networks. Sakai*et al.* [17] construct simplified mathematical models to understand the fundamental performance and security guarantees of onion-based anonymous routing in delay tolerant networks (DTNs). Zhao *et al.* [18] presents an evaluation model to describe the mutual influence of network security and QoS. In this model, the main security factor is the security mechanism. The authors use the security strength to describe the security mechanism and its resource consumption and evaluate its negative impact on systems QoS. Nieto *et al.* [19] provide a context-based parametric relationship model (CPRM) to help system administrators address security and QoS tradeoff issues in the configuration of the environment. CPRM can analyze the security and QoS tradeoff from a parametric point of view. Security mechanisms as the main security factors are considered in CPRM. Aldini and Bernardo [20] present an approach to predict the tradeoff between security and QoS, and use model checking to verify the equivalences between both specifications to control illegal information flows in the system. In [21], a modeling approach is used to provide a context-based model to balance QoS and security. However, the impact of malicious attacks on performance is not systemically addressed in these studies. Wang *et al.* [22] constructs a multiple queueing model to characterize three types of attacks integrally, and study the performance metrics of specific email system security. However, the resource consumption of security mechanisms has not been addressed thoroughly. Chen *et al.* [23] create a pricing mechanism for on-demand security as a service for the cloud-enabled internet of controlled things (IoCT). A contract-based FlipCloud game is used to assess the security risk and the cloud QoS under advanced persistent threats in this approach. However, the cost that describes the impact of the attack on the QoS is difficult to quantify, so the results in [23] do not accurately reflect the impact of attacks on the Qos.

In this paper, a hierarchical modeling approach is proposed to build the connection between the security and the service performance. First, a Markov state transition model is built to evaluate VM security. Two security factors, the security mechanism and the malicious attack, are considered in this model. Second, queue theory is used in the performance modeling of the CCS. Then, the Bayesian approach is applied to connect the security and the service performance and obtain the complex S-P correlations. Finally, the correlation metric is proposed.

The contributions of this paper are summarized as follows:

1) A VM security evaluation model in CCS is proposed, which considers two security factors including the security mechanism and the malicious attack.
2) A hierarchical S-P modeling approach in CCS is proposed.
3) A correlation metric that considers the impact of security on the service performance is proposed.
4) A large set of parameters including the number of VMs, the intensity of the security mechanism in the VMs, the available CPU resources, the request arrival rate and the service rate are considered in this approach to improve its fidelity.
5) The proposed model covers many failures that may occur in the servicing process of the CCS including VM failure, time-out failure and overflow failure.
6) The Bayesian approach is adopted in this approach.

The rest of this paper is organized as follows. Section 2 gives an overview of the physical and logical structure of the CCS. Section 3 introduces the security and the service performance modeling approaches and proposes the correlation metric. In Section 4, experiments are made to verify the hierarchical modeling approach and reveal the impact of different intensities of security mechanisms on the service performance. Finally, a summary and future research plans are provided in Section 5.

II. CLOUD SYSTEM AND CORRELATION METRIC

In this section, the physical and logical structure of the CCS are first described. Then, the impact of the security on the service performance is briefly introduced.

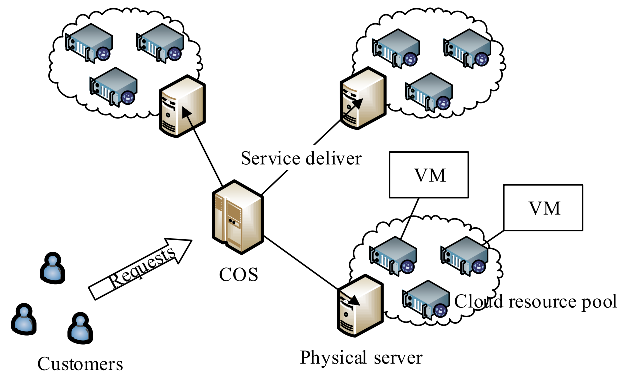


FIGURE 1. **General physical architecture of the CCS.**

A. THE PHYSICAL ARCHITECTURE OF THE CCS

Fig. 1 shows the general physical architecture of the CCS. The most essential component of the CCS is the cloud operating system (COS). The COS works on the infrastructure level and manages the virtual infrastructure (such as virtual machines, the virtual switch, and so on) as well as the back-end hardware and software resources. In addition, the COS is in charge of processing user service requests. In the executing process, the service request is first translated into several subtasks. Then, those subtasks are sent to suitable VMs for execution. Finally, results are sent back from VMs to the COS. The service request is finished only if all subtasks are processed and all correct results are sent back to the COS. Obviously, VMs have a great impact on service processing.

B. THE LOGICAL STRUCTURE OF THE CCS AND THE CORRELATION BETWEEN SECURITY AND PERFORMANCE

From physical architecture to logical structure, Fig. 2 shows the two-layer logical structure of the CCS. In fact, there are more than two logical layers in the CCS. Only the application layer and resource layer are related to this paper and shown in Fig. 2.

The most basic layer in the CCS is the resource layer, which contains all the hardware and software including the storage, server infrastructure, and virtual infrastructure. In the resource layer, the CCS organizes resources in units of VM that are a highly automatic system on the CCS. The VM usually is individually accessible over the internet [24], [25]. This feature makes the VM vulnerable to malicious attacks. For example, the Common Vulnerability and Exposures (CVE) initiative identifies a vulnerability (CVE-2007-4593) that enables an attacker to launch a denial of service (DoS) attack on the virtual machine ware (VMWare) Workstation 6.0 via vstor2-ws60.sys device driver [26]. DoS attacks can slow down some critical processes and make failure happen. Obviously, VM security is an unavoidable vulnerability for the CCS. So, evaluating VM security including learning the threaten from malicious attacks and analyzing the action of defending mechanism can helps us improve cloud security or even increases service performance. It is noteworthy that malicious attacks are random. Theoretical modeling approach can just catch this randomness depending on its characteristic of abstraction and the view of macroscopic statistics. As we know, Markov model theory has been wildly used in security evaluation field. For example, Kreidl O P [27], Wang *et al.* [22] and Madan *et al.* [28] use it in their research. Therefore, a Markov model is built to evaluate VM security in this paper.

In the application layer, the COS is in charge of the allocation of service requests and the collection of results. The number of requests returned from the COS to users can be a metric for describing the service performance of the CCS. Therefore, the service performance is defined as the number of accomplished requests per unit time in this paper. As mention above, The VM has a great impact on service processing. On the one hand, when the availability of VMs are threaten by malicious attacks, VMs may fail to process subtasks and result in the processing failure of subtasks. More seriously, malicious attacks may reduce the number of available VMs and result in the overflow failure that occurs when the available VMs are too few. On the other hand, application of security mechanisms in the VM consumes a part of CPU resources and decreases the CPU resources that are used to process subtasks. Eventually, it may extend the processing time of the subtasks and result in overtime failure. In general, security affects the service performance by affecting the number of available VMs and the CPU processing ability of a VM. Therefore, those two parameters can be the bridge between security and performance.

As we all know, the CPU occupation of a security mechanism depends on its algorithm complexity or security level, such as the encryption mechanism and the hierarchical protection domains. For the encryption mechanism, the length of key represents the algorithm complexity and the security level of encryption mechanism and determines the CPU occupation time of the action including encryption and decryption. For hierarchical protection domains, the security level determines the number of rings and the CPU occupation time for switching between user states and kernel states. To better describe the algorithm complexity or security level, intensity has been used in the contemporary dependability and security research field [12], [29]–[31]. Therefore, intensity is also used to define the algorithm complexity or the security level

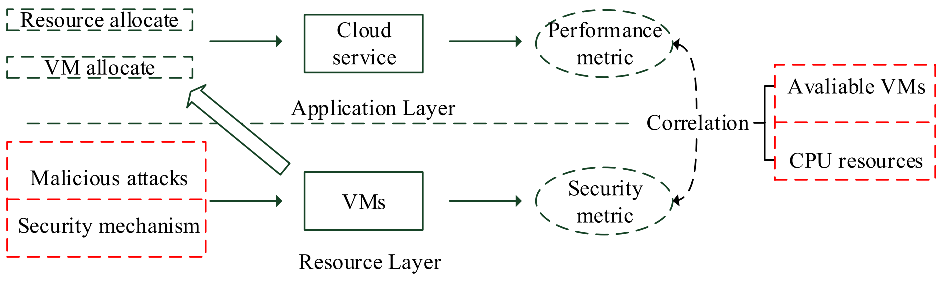


FIGURE 2. Two-layer logical structure of the CCS.

of security mechanism in this paper. Three intensities are defined for security mechanism including high, middle, and low. The higher the intensity, the higher the security level and the more complex the algorithm are. If the VM applies the high-intensity security mechanism, the completing time of subtask are increased comparing with applying the middle-intensity or low-intensity security mechanism. It is noteworthy that, in addition to the CPU, some other resources also have effect on performance, such as memory allocation and storage allocation. But, we only consider CPU in this paper.

In next section, the security and performance modeling approaches are specifically introduced. The correlation metric is also given.

III. MODELING APPROACHES

In this section, the security modeling approach and performance modeling approach are described including their establishment conditions, hypothesis, structure, parameters, analysis metric, and model solving process. Then, the correlation metric is proposed depending on those two approaches.

A. SECURITY MODELING APPROACH IN RESOURCE LAYER

1) ESTABLISHMENT CONDITIONS

The Markov model can be used to model VM security for the following reasons.

1) Most attacks are unorganized and spontaneous. Therefore, the arrival time of the attack is random. This randomness can be described by a suitable arrival process, such as a Poisson distribution [22], [27], [28], [32].
2) Security mechanisms have a series of normalized operations to defend against attacks. According to existing approaches [22], [27], [28], the time of those operations obey the exponential distribution.

Note that the distribution functions of sojourn time for some states may be non-exponential in the Markov model. Therefore, the underlying stochastic model needs to be formulated in terms of a semi-Markov process (SMP) [28].

2) MODEL VM SECURITY

Fig. 3 shows the state transition diagram of VM under attack. There are three states, good state (G), under attack state (A), and fail state (F) in this state transition diagram.

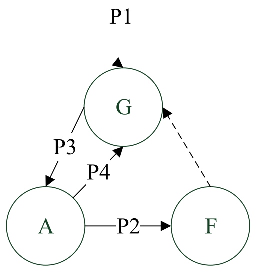


FIGURE 3. The state transition diagram of a VM under attack (an embedded discrete time Markov chain for the SMP model).

State G represents the good status of the VM in which it can provide dependable service. State A represents the status of the VM affected by attack. However, it can still provide service. State F represents the failed status of VM under attacks, which cannot provide service. In Fig. 3, p_1, p_2, p_3, and p_4 are the state transition probabilities. Their values are determined by the intensity of the security mechanism and the attacks. p_1 represents the probability that the VM remains in G. p_2 represents the probability that the status of the VM changes from A to F under the circumstances that the security mechanism fails against attacks. p_4 represents the probability that the status of the VM changes from A to G when the security mechanism successfully resists attacks. $p_4 = 1 - p_2$. p_3 represents the probability that the status of the VM changes from G to A when the attacks have affected VM. $p_3 = 1 - p_1$. In addition, the situation that the status of the VM changes from G to F is not considered. This is because cloud service providers generally pay much attention to security, which makes it difficult for attacker to invalidate the VM without being noticed. Note that state F is an absorbing state, which means that one cannot change to any other state from this state. In a real environment, the security mechanism may not be strong enough to defend against attacks. When this happens, the system cannot return to normal unless the system administrator takes emergency actions, such as, rebooting, cutting off internet access, and so on. Therefore, a dotted line is used to describe the situation in which the status of the VM changes from F to G after activating those emergency actions.

Now, the VM security evaluation can be performed by solving this state transition diagram. Two security metrics including $N_a(x, n)(x \leq n)$ and p_a are defined, in which n is the total number of VMs and x is the number of available VMs. $N_a(x, n)$. represents the probability that there are x VMs available among n VMs. p_a represents the probability that the VM is available. $N_a(x, n)$ and p_a are used to evaluate the security of the entire system and one VM separately. Notably, p_a is the precondition for calculating $N_a(x, n)$.

In the following, the solving process of p_a is given. First, the status of the VM is divided into an available set, x_a, and an unavailable set, x_ua. If the status of the VM belongs to x_a, this VM is available. Otherwise, the VM is unavailable, belonging to x_ua. In this paper, G and A are classified as set x_a and F is classified as set x_ua. Second, π_G, π_A, and π_F are defined as the steady-state probabilities of states G, A, and F, respectively. $\sum_{i\in(G,A,F)} \pi_i = 1$. Then, p_a can be obtained as

$$p_a = \pi_G + \pi_A = 1 - \pi_F \tag{1}$$

Applying the method introduced in [28] to solve the SMP model shown in Fig. 3, the $\pi_i(i \in (G, A, F))$ is calculated as

$$\pi_i = \frac{v_i h_i}{\sum_i v_i h_i} \tag{2}$$

In (2), $v_i(i \in (G, A, F))$ is the steady-probability of the state in the embedded discrete time Markov chain (DTMC) shown in Fig. 3 and $h_i(i \in (G, A, F))$ are the mean sojourn time of each state. v_i is obtained as

$$\bar{v} = \bar{v} \cdot P \tag{3}$$

where $\bar{v} = [v_G, v_A, v_F]$ and P is a transition probability matrix. P equals

$$\begin{array}{c|ccc} & G & A & F \\ \hline G & P_1 & 1-P_1 & 0 \\ A & 1-P_2 & 0 & P_2 \\ F & 1 & 0 & 0 \end{array}$$

Obtaining the value of P and h_i is a problem. There are two methods for estimating these values. One method is to use a priori knowledge and attack experiments which are introduced in [33] and [34]. The other is using parameter training method which depends on the predefine p_a and model (note that this method is chosen for use in experimental section of this paper). Here, it is worthwhile to briefly explain the second method. In some cases, a priori knowledge of security mechanisms and attacks is limited. Only the data for the number of VM failures and the time of attack can be obtained from system logs over a period of time. Then, the p_a can be calculated using those data and probability theory. Finally, the value of P and h_i can be obtained through p_a and a parameter training method.

Now, p_a is solved. As mention before, p_a is the precondition for calculating $N_a(x, n)$. If x and n are assigned and the p_a of each VMs is known, then there are C_n^x possible combinations for the available VMs group. Then, according to Bayesian theory, $N_a(x, n)$ is solved by summing the probability of each combinations.

We use the calculation of $N_a(3, 5)$ as an example. $N_a(3, 5)$ means that there is total of 5 VMs (No.1 to No.5) in the CCS in which only 3 are available. Malicious attack makes the other 2 unavailable. In this case, we do not know which three of them are available. In fact, each combination is equally likely to occur. For example, one possible combination is that No.1, No.2, No.3 are available, whereas No.4, No.5 are unavailable. Another combination is that No.2, No.3, No.4 are available, whereas No.1, No.5 are unavailable. Therefore, all possible combinations should be considered to calculate $N_a(3, 5)$. After summing the probability of all possible combinations and calculating the averaging results, $N_a(3, 5)$ is obtained. A numerical example is as following.

p_a^i is defined as the probability that i VM is available. Then, the function that is used to describe the combination that No. 1, No.2, No. 3 are available and No. 4, No. 5 are unavailable is

$$N_a^1 = p_a^1 \cdot p_a^2 \cdot p_a^3 \cdot (1 - p_a^4) \cdot (1 - p_a^5)$$

The functions of other combinations are computed in the same way. Summing the results of those combinations and dividing this sum by the number of those combinations (C_5^3), the value of $N_a(3, 5)$ is obtained. The result is shown below

$$N_a(3, 5) = \frac{N_a^1 + N_a^2 + N_a^3 + \ldots + N_a^{C_5^3}}{C_5^3}$$

Applying this method, $N_a(x, n)$ is obtained. The next section introduces the performance modeling.

B. PERFORMANCE MODELING APPROACH IN THE APPLICATION LAYER

1) ESTABILISHMENT CONDITIONS

Queuing theory has been widely used to model the arriving-completing process of random events in a real environment. In this paper, an M/M/S queuing model is used to describe the arriving-completing process of tasks in the CCS. M/M/S means that the arrival of tasks follows the Poisson distribution, the service time of sub-tasks follows the exponential distribution, and the queue length of tasks is limited.

To facilitate computing, we assume that

1) All VMs in the same trust zone (TZ) which is a combination of network segmentation [1] apply the unitive security mechanism that have the same intensity. In real cloud environment, the CCS consists of numerous interconnected TZ [1]. TZ is set up to facilitate the management and processing of different types of tasks, in which tenant applications often across the same type of VM instances [35]. Therefore, assuming that all VMs apply the unitive security mechanism in the same TZ is workable.
2) VM and CPU obey a one-to-one logical mapping mechanism, which is very common in the CCS.
3) In VM, only one task is executed at one time.
4) The task queue obeys the rule of first come first serve (FCFS).

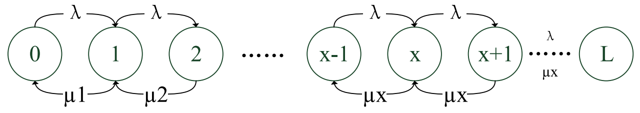


FIGURE 4. The birth-death process of the task queue.

5) When all available VMs are busy, the newly arrived tasks should wait in the task queue. If the number of tasks in the task queue equals the maximum length of the task queue, the newly arriving task is abandoned.

2) PERFORMANCE MODELING

Fig. 4 gives the birth-death process of the task queue. Parameters of this task queue are defined as following.

1) f_j^d is the CPU occupation rate of the running security mechanism in VM j.
2) λ is the arrival rate of tasks, which can be defined as the number of arriving tasks per unit time. For example, $\lambda = 2.1s^{-1}$.
3) f_j is the available CPU rate excepting the part occupied by the security mechanism in VM j. And, $f_j \leq 1 - f_j^d$.
4) n is the number of VMs, which contains the number of all available and unavailable VMs.
5) x is the number of available VMs, $x \leq n$.
6) L is the maximum length of the task queue.
7) μ is the maximum service rate of the VM without applying any security mechanism, which can be defined as the number of completed tasks per unit time. For example, $\mu = 1.1s^{-1}$.
8) μ_i is the service rate when i tasks are executing at the same time, $i \leq L$. According to assumption 1, all VMs apply the unitive security mechanism in one TZ. Then, μ_i can be calculated using

$$\mu_i = i \cdot \frac{\sum_{j=1}^{n} \mu(f_j)}{n}, \quad i \leq x \tag{4}$$

$$\mu_i = \mu_x = \sum_{j=1}^{x} \mu(f_j), \quad x \leq i \leq L \tag{5}$$

$\frac{\sum_{j=1}^{n} \mu(f_j)}{n}$ represents the average service rate of each VM in TZ. $\mu(f_j)$ is the service rate of VM j, which is a function of parameter f_j and μ. $\mu(f_j)$ represents the ability of processing task of VM ðİ′− when some part of the CPU is occupied by the running security mechanism.

9) T_d is the time of the task. If the sojourn time of the task exceeds its due time T_d, a time-out failure occurs.

The performance of the CCS can be evaluated by analyzing this task queue model. Here, the performance metric is defined as $\delta(x)$ with x representing the number of available VMs. $\delta(x)$ gives the number of completed tasks per unit time. Task completion means that a time-out failure has not happened during the entire execution process of the task. According to queue theory, $\delta(x)$ can be obtained as

$$\delta(x) = \eta_e(1 - p_{time-out}) \tag{6}$$

$(1 - p_{time-out})$ represents the probability that the task has been successfully completed and the time-out failure has not happened during entire execution process of the task. η_e denotes the effective arrival rate of new tasks. Effective arrival means that the task has been placed in the task queue without being abandoned. Therefore, η_e is given as

$$\eta_e = \lambda(1 - q_L) \tag{7}$$

As mentioned above, λ is the arrival rate of tasks. q_L represents the probability that the task queue is full. When the queue is full, a new arriving task is abandoned which means that a blocking failure occurs. To calculating q_L, q_i is first defined as the probability that there are i tasks in the task queue. It is easy to derive q_i by building and solving the Chapman Kolmogorov equations for Fig. 4. Then, q_i is given as

$$q_0 = [1 + \sum_{i=1}^{x-1} \frac{\lambda^i}{\prod_{r=1}^{i} \mu_r} + \sum_{i=x}^{L} \frac{\lambda^i}{\mu_x^{i-x} \prod_{r=1}^{x} \mu_r}]^{-1} \tag{8}$$

$$q_i = \frac{\lambda^i}{\prod_{r=1}^{i} \mu_r} q_0, \quad (1 \leq i < x) \tag{9}$$

$$q_i = \frac{\lambda^i}{\mu_x^{i-x} \prod_{r=1}^{x} \mu_r} q_0, \quad (x \leq i \leq L) \tag{10}$$

According to (10), q_L is obtained as

$$q_L = p_{block} = \frac{\lambda^L}{\mu_x^{L-x} \prod_{r=1}^{x} \mu_r} q_0 \tag{11}$$

Thus, η_e is solved. For $(1 - p(time - out))$, $p(time - out)$ is the probability that the executing time of the task has exceeded the due time T_d. According to the definition of $p(time - out)$, it can be calculated as

$$p_{time-out} = Pr\{T_s > T_d\} = 1 - \int_0^{T_d} f_s(t)dt \tag{12}$$

T_s represents the sojourn time of the task, which is the processing time of the task. $f_s(t)$ is defined as the probability density function (pdf) of the sojourn time T_s. Obviously, T_s is the sum of the tasks execution time T_e and waiting time T_w. Therefore, $f_s(t)$ can be obtained as

$$f_s(t) = \sum_{i=0}^{x} Pr(i) \cdot f_e(t) + \sum_{i=x+1}^{L-1} Pr(i) \cdot f_{i-x+1}(t) \bigotimes f_e(t) \tag{13}$$

where $\bigotimes$ represents the convolution operator of two function. $Pr(i)$ is the probability that when a new task has arrived, there are i tasks in the task queue. According to the Bayesian approach and (8)(9)(10), $Pr(i)$ can be calculated as

$$Pr(i) = Pr\{i|i < L\} = \frac{Pr\{i, i < L\}}{Pr\{i < L\}} = \frac{q_i}{1 - q_L}, \quad (i = 0, 1, \ldots, L - 1) \tag{14}$$

As mentioned above, the service time of the task follows an exponential distribution. $f_e(t)$ is defined as the pdf of the execution time T_e with parameter μ_i. $f_e(t)$ is given as

$$f_e(t) = \mu_i \cdot e^{-\mu_i t} \tag{15}$$

The pdf of the waiting time T_w is given as $f_(i-x+1)(t)$. As known from Fig. 4, T_w for task i is the processing time of the $i-x+1$ tasks in front of it. Obviously, T_w obeys an Erlang distribution. Then, $f_(i-x+1)(t)$ can be obtained as

$$f_{i-x+1}(t) = \frac{(x\mu_x \cdot t)^{i-x}}{(i-x)!} \cdot x\mu_x \cdot e^{-x\mu_x \cdot t}, \quad t \geq 0,\ x < i < L \tag{16}$$

Thus, $(1-p_(time-out))$ is solved. Substituting η_e and $(1-p_(time-out))$ into (6), $\delta(x)$ is obtained. In the next section, the correlation metric is proposed.

C. THE CORRELATION METRIC

As mentioned before, on the one hand, the security metric ($N_a(x,n)$) reflects the impact of security factors on the number of available VMs. On the other hand, the performance metric ($\delta(x)$) reflects the impact of the number of available VMs on the service performance. Spontaneously, the impact of security factors on the service performance can be evaluated by combining $N_a(x,n)$ with $\delta(x)$. Thus, the correlation metric is proposed to reflect this impact. $E(\delta(x))$ is defined as the correlation metric, which represents the new performance metric taking the impact of security into consideration. According to the Bayesian approach, $E(\delta(x))$ is calculated as

$$E(\delta(x)) = \sum_{x=1}^{n} N_a(x,n) \cdot \delta(x) \tag{17}$$

The number of available VMs (x) are uncertain in the CCS, and can equal any number between 1 and n with a probability of $N_a(x,n)$. Therefore, $E(\delta(x))$ is obtained by summing over all these probabilities. Numerical examples are introduced in section 4.

IV. EXPERIMENT

In this section, a simulation experiment is first taken to verify modeling approaches by using matrix laboratory (MATLAB), in which the simulation results calculated using the simulation program are compared with the theoretical results calculated using the modeling approach. Then, the experiments that reveal the impact of different intensities of security mechanisms on the service performance are conducted. Finally, three optimization recommendations about how to efficiently improve service performance while maintaining security to some extent are proposed.

A. VERIFICATION

Tables 1 $\sim$ 4 show the assignment of parameters for the security modeling approach and the performance modeling approach. Introductions to the assignment of parameters are given in the next paragraph.

TABLE 1. Parameter value settings for different intensity security mechanism.

Intensity of security mechanism	The available probability of VM	State transition probability		Mean sojourn time		
	p_a	P_1	P_2	h_G	h_A	h_F
high	0.84	0.8	0.4	3	2	8
middle	0.75	0.7	0.43	2.6	1.6	8
low	0.66	0.62	0.48	2.4	1.4	8

TABLE 2. The CPU consumption rate of different intensity security mechanism.

Intensity of security mechanism	f_j^d
low	2%
middle	21%
high	35%

TABLE 3. Parameter value settings in performance model.

parameters	value
λ	$2.1s^{-1}$
L	12
T_d	$7s$
μ	$0.7s^{-1}$
n	4

TABLE 4. Special parameter definition in performance model.

parameters	definition
f_j	$f_j = 1 - f_j^d$
$\mu(f_j)$	$\mu(f_j) = \mu \cdot f_j$

As mentioned in the section of security modeling approach, the parameter training method is used to obtain the value of P and h_i. Therefore, the value of p_a is assigned directly in table 1. It is noteworthy that many parameter training methods have been well studied. Here, we don not introduce the details of the parameter training method. In table 2, the value of f_j^d is assigned, following Batista *et al.* [12]. In fact, obtaining the CPU occupation rate of the security mechanism is easy in real environment. For example, contemporary operating systems offer a service to detect the CPU occupation rate of each running applications. In table 3, the value of parameter x is not assigned. Instead, it depends on n according to the definition of the correlation metric. Popa *et al.* [36] and Zaman *et al.* [11] indicated that the security and the service performance are inversely proportional quantities. Therefore, the parameter definitions are given in table 4 to satisfy the correlation between the security and the service performance.

In the following, some important definitions about simulation are given.

1) μ changes from 0.5 to 2.6 in 0.1interval.
2) All 4 VMs apply a high-intensity security mechanism.
3) The simulation is executed 50*21 times. In each iteration, a main loop structure that loops 1000 times is defined. In the simulation, one loop means one unit time in theoretical modeling approach. 50*21 iterations will produce 50*21 outputs. Each of these outputs

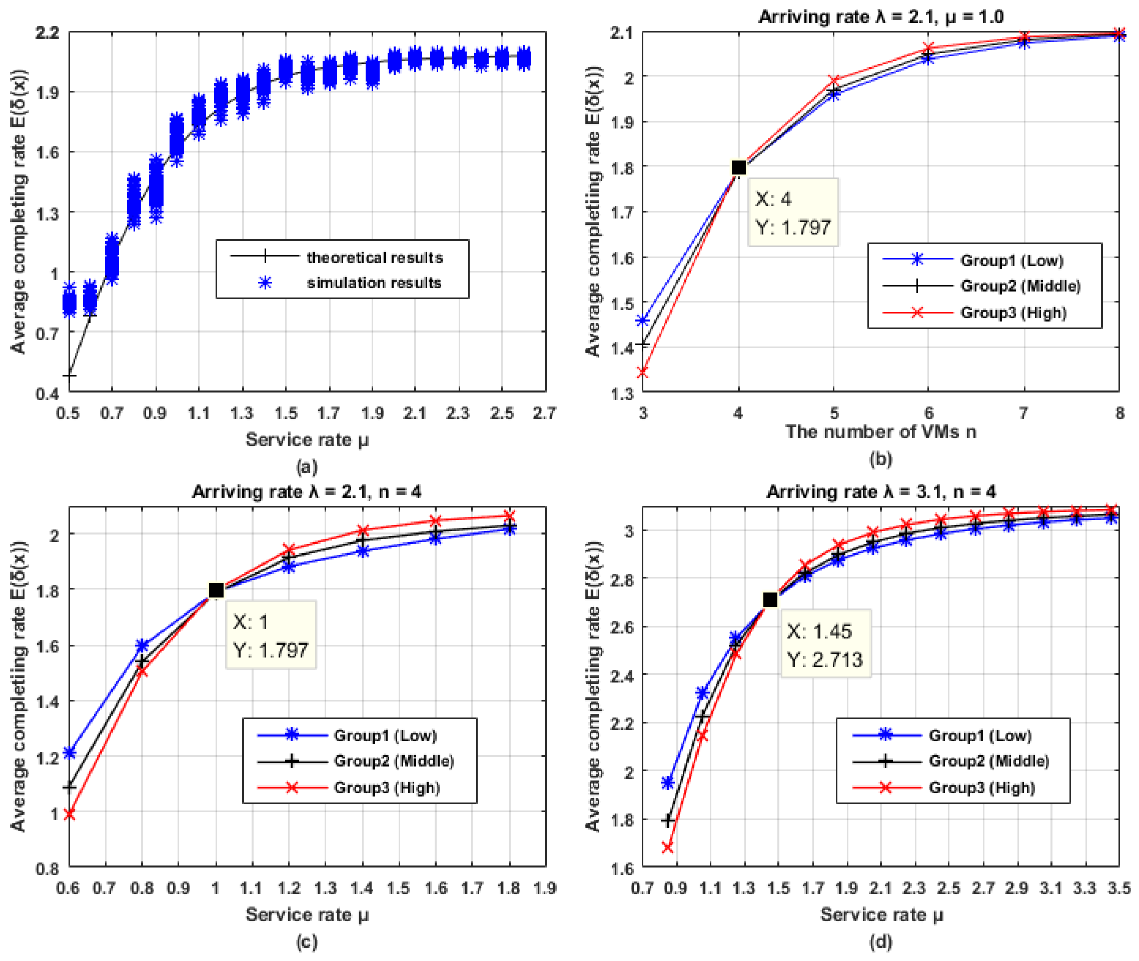


FIGURE 5. **(a) The comparison between theoretical and simulation results; (b) Plot of $E(\delta(x))$ against VMs n from 3 to 8; (c) Plot of $E(\delta(x))$ against the service rate μ from 0.6 to 1.8; (d) Plot of $E(\delta(x))$ against the service rate μ from 0.8 to 3.4.**

represent the completion rate of tasks under different values of μ. The value of 50 means that 50 results are produced for each value of μ. The value of 21 means that there are 21 different value for μ.

4) Each task can be completed during one loop (one unit time).
5) If the VM is in absorbing state F, it will change from F to G after h_F unit of time. h_F is assigned as shown in table 1.
6) Information for each task including the number of arrived, accomplished, and abandoned tasks is recorded.
7) A random number generator is set. In a simulation, a probability (the available probability of the VM) has been predefined to decide the availability of each VMs. In each loops, the availability of each VMs is determined by comparing the pre-define probability with a random number.
8) λ determines the number of new arriving tasks per unit of time. μ, f_j^d, and the number of available VMs jointly determine the number of completed tasks per unit of time.

The comparison results are shown in Fig. 5(a). In Fig. 5(a), the theoretical results perfectly match simulation results with the increase of the service rate. This result proves the correctness of the theoretical modeling approach proposed in this paper. In the next section, the experiments that reveal the impact of different intensities of security mechanisms on the service performance are conducted.

B. PERFORMANCE ANALYSIS BASED ON DIFFERENT PARAMETERS SETTING

In this section, three independent variables, the number of VMs (n), the CPU occupation rate of the security mechanism (f_j^d), and the service rate of VM (μ), are studied to reveal their impact on service performance. To reflect the impact of the CPU occupation rate of the security mechanism (f_j^d) on the service performance, three comparing groups, groups 1, 2, and 3, are designed. Groups 1, 2, and 3 adopt low, middle, and high intensity security mechanisms, respectively. As mentioned before, each groups consists of the same number of VMs that adopt the same intensity security mechanism.

In the first experiment, the value of μ and n are changed to reveal their impact on the service performance. The assignments of μ and n are as follows

1) For μ, two groups of values are designed. The first group of value for μ ranges from 0.6 to 1.8, and the arrival rate $\lambda = 2.1s^{-1}$. The second groups of value for μ ranges from 0.8 to 3.4 and the arrival rate $\lambda = 3.1s^{-1}$. In these two experiments, the number of VMs is $n = 4$. (The results are shown in Figs. 5 (c) and (d)).
2) For n, the value ranges from 3 to 8. In this experiment, the service rate is $\mu = 1.0s^{-1}$. (The results are shown in Fig. 5 (b)).
3) The values of the CPU occupation rate of different security mechanisms (f_j^d) are the same as those in table 2.
4) The assignment of other parameters setting are the same as those in table 1, 3, and 4.

The results are plotted in Fig. 5. The abscissa represents the service rate of the VM (in Fig. 5(c) and Fig. 5(d)) and the number of VMs (in Fig. 5(b)). The ordinate represents the average completed rate of tasks $E(\delta(x))$. The results of this experiment are analyzed as follows.

1) When the amount of available resources including the service rate of the VM (μ) and the number of VMs (n) are limited, there is an inverse correlation between the service performance and the security. For example, when $\mu = 0.6 \sim 1.0$ in Fig. 5(c) or $n = 3 \sim 4$ in Fig. 5(b), the values of $E(\delta(x))$ for a low-intensity security mechanism (blue line, group1) are bigger than the values of $E(\delta(x))$ for a middle-intensity security mechanism (black line, group2) and a high-intensity security mechanism (red line, group3). These results reveal that high security is not always the best choice at any moment.
2) With the increase of available resources, there is a positive correlation between the service performance and the security. As shown in Fig. 5, the rate of increase of $E(\delta(x))$ for a high-intensity security mechanism (red line) and middle-intensity security mechanism (black line) are faster than the rate of increase of $E(\delta(x))$ for low-intensity security mechanism (blue line).
3) Redundancy is a good strategy for improving the service performance. In Fig. 5(b), the value of $E(\delta(x))$ rapidly increases with the increase in the number of VMs (n).

In the second experiment, a three dimensional (3D) map of $E(\delta(x))$, the number of VMs (n) and the service rate of the VM (μ) is plotted. The assignments of f_j^d, μ, and n are as follows

1) n ranges from 2 to 8. All VMs apply high-intensity security mechanism ($f_j^d = 35\%$). The same results would be obtained using another security mechanism with a different intensity.
2) The service rate μ ranges from 0.8 to 1.4.
3) The assignment of other parameters are same as in tables 1, 3, and 4.

The results are plotted in Fig. 6. The color change from purple to yellow in Fig. 6 represents the size of $E(\delta(x))$. The greater the value of $E(\delta(x))$ is, the closer the color is to yellow, and vice versa.

Fig. 6 shows that the impact of the number of VMs on the service performance is greater than the service rate. The change rule of $E(\delta(x))$ is shown in Fig. 6. For example, fixing one of two parameters, including the number of VMs ($n = 3$) or the value of the service rate ($\mu = 0.9s^{-1}$), and changing another parameter observes the change of $E(\delta(x))$. Obviously, increasing the number of VMs brings more rapid growth for $E(\delta(x))$.

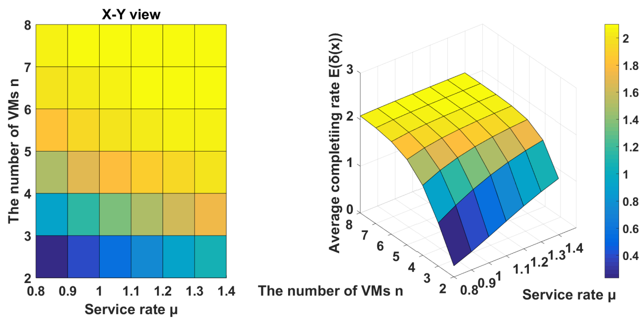


FIGURE 6. The $E(\delta(x))$ under different parameter settings.

In these experiments, it is assumed that VMs in the same group use the same intensity security mechanism. If VMs in the same group can use different intensity security mechanisms, the results demonstrated from these experiments are still valid. In fact, this is easy to infer. The first experiment demonstrates that with an increase of the service rate, the revenue earned by applying a high-intensity or middle-intensity security mechanism is greater than that earned by applying a low-intensity security mechanism. This result means that when resources, including the CPU processing power and the number of VMs, are large enough in CCS, improving the average security for the entire CCS can improve the service performance of the CCS. Therefore, the service performance curve of the general situation (in which VMs in the same group can use different intensity security mechanisms) must be in the region between the curve of the group that applies a low-intensity security mechanism and the group that applies high-intensity security mechanism. For example, in the first experiment, we define a new comparing group, group4 in which one VM applies a low-intensity security mechanism and three VMs apply a middle-intensity security mechanism. Obviously, the performance curve of group 4 is between the curve of group1 and group2.

In general, these two experiments demonstrate the impact of security on the service performance. Specifically, two security factors, the malicious attack and the security mechanism affect the service performance by affecting the service rate of the VM and the number of available VMs. Therefore, security is an indispensable factor for service performance evaluation. In the next section, three optimization recommendations are summarized according to the results from these two experiments.

C. OPTIMIZATION RECOMMENDATIONS

In the previous section, some valuable conclusions are obtained by analyzing the first and second experiment. The following optimization recommendations about balancing the security and the service performance are now proposed.

1) When the amount of available resources (such as, the CPU and the number of VMs) are limited in CCS, a lower-intensity security mechanism may bring a better service performance.
2) Deploying more VMs improves the service performance faster than improving the CPU processing power of each VM.
3) To simultaneously improve the security and the service performance, applying a powerful CPU with high processsing power is better than deploying more VMs.

V. CONCLUSION

In this paper, a hierarchical modeling approach and a correlation metric are proposed to study the impact of the security (i.e., malicious attack and security mechanism) on the service performance. The presented modeling approach builds the connection between the security and the service performance through two critical factors (i.e., the number of VMs and the service rate). Simulation experiments verify the correctness of the presented modeling approach. Moreover, the significant impact of different factors, including the security mechanism, service rate, and the number of VMs, on the performance are analyzed through experiments. Experimental results demonstrate that: 1) When the CCS faces serious security issues, redundancy is the easiest and most workable way to improve the service performance. 2) High security is not always the best choice at any moment. 3) When the amount of resources is limited, the correlation between the service performance and the security is inverse. Otherwise, the correlation between the service performance and the security is positive. In future work, we will explore the relevant optimal scheduling algorithms based on this modeling approach to better support service-oriented computing.

REFERENCES

[1] D. Gonzales, J. M. Kaplan, E. Saltzman, Z. Winkelman, and D. Woods, "Cloud-trust—A security assessment model for infrastructure as a service (IaaS) clouds," *IEEE Trans. Cloud Comput.*, vol. 5, no. 3, pp. 523–536, Jul./Sep. 2017.

[2] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds," in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, 2009, pp. 199–212.

[3] M. Schwartz, "New virtualization vulnerability allows escape to hypervisor attacks," *Inf. Week Secur.*, vol. 4, p. 12, 2013.

[4] E. Ray and E. Schultz, "Virtualization security," in *Proc. 5th Annu. Workshop Cyber Secur. Inf. Intell. Res., Cyber Secur. Inf. Intell. Challenges Strategies*, 2009, Art. no. 42.

[5] F. Breedijk. (Jul. 31, 2009). *Blackhat Talk: Cloudburst–VMWare Guest to Host Escapes by Kostya Kirtchinsky*. [Online]. Available: http://www.cupfighter.net/index.php/2009/07/blackhat-cloudburst-vmware-guest-to-host-escape:/

[6] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 843–859, 2nd Quart., 2013.

[7] K. Hwang, X. Bai, Y. Shi, M. Li, W.-G. Chen, and Y. Wu, "Cloud performance modeling with benchmark evaluation of elastic scaling strategies," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 130–143, Jan. 2016.

[8] J. Cao, K. Hwang, K. Li, and A. Y. Zomaya, "Optimal multiserver configuration for profit maximization in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1087–1096, Jun. 2013.

[9] K. Hwang, Y. Shi, and X. Bai, "Scale-out vs. Scale-up techniques for cloud performance and productivity," in *Proc. IEEE 6th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Dec. 2014, pp. 763–768.

[10] A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "Performance analysis of cloud computing services for many-tasks scientific computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 6, pp. 931–945, Jun. 2011.

[11] S. Zaman, B. Adams, A. E. Hassan, "Security versus performance bugs: A case study on Firefox," in *Proc. 8th ACM Work. Conf. Mining Softw. Repositories*, 2011, pp. 93–102.

[12] B. G. Batista, C. H. G. Ferreira, D. C. M. Segura, D. M. L. Filho, and M. L. M. Peixoto, "A QoS-driven approach for cloud computing addressing attributes of performance and security," *Future Gener. Comput. Syst.*, vol. 68, pp. 260–274, Mar. 2017.

[13] K. Wolter and P. Reinecke, "Performance and security tradeoff," in *Formal Methods for Quantitative Aspects of Programming Languages*. Springer-Verlag, 2010, pp. 135–167.

[14] S. S. Yau, Y. Yin, and H. G. An, "An adaptive tradeoff model for service performance and security in service-based systems," in *Proc. IEEE Int. Conf. Web Services*, Jul. 2009, pp. 287–294.

[15] W. Zeng and M. Y. Chow, "Modeling and optimizing the performance-security tradeoff on D-NCS using the coevolutionary paradigm," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 394–402, Feb. 2013.

[16] J.-H. Cho, I.-R. Chen, and P.-G. Feng, "Performance analysis of dynamic group communication systems with intrusion detection integrated with batch rekeying in mobile ad hoc networks," in *Proc. IEEE 22nd Int. Conf. Adv. Inf. Netw. Appl.-Workshops*, Mar. 2008, pp. 644–649.

[17] K. Sakai, M.-T. Sun, W.-S. Ku, J. Wu, and F. S. Alanazi, "Performance and security analyses of onion-based anonymous routing for delay tolerant networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 12, pp. 3473–3487, Dec. 2017.

[18] X. Zhao, Q. Lin, J. Chen, X. Wang, J. Yu, and Z. Ming, "Optimizing security and quality of service in a real-time database system using multi-objective genetic algorithm," *Expert Syst. Appl.*, vol. 64, pp. 11–23, Dec. 2016.

[19] A. Nieto and J. Lopez, "A context-based parametric relationship model (CPRM) to measure the security and QoS tradeoff in configurable environments," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 755–760.

[20] A. Aldini and M. Bernardo, "A formal approach to the integrated analysis of security and QoS," *Rel. Eng. Syst. Saf.*, vol. 92, pp. 1503–1520, Nov. 2007.

[21] M. Alia, M. Lacoste, R. He, and F. Eliassen, "Putting together QoS and security in autonomic pervasive systems," in *Proc. 6th ACM Workshop QoS Secur. Wireless Mobile Netw.*, 2010, pp. 19–28.

[22] Y. Wang, C. Lin, and Q.-L. Li, "Performance analysis of email systems under three types of attacks," *Perform. Eval.*, vol. 67, pp. 485–499, Jun. 2010.

[23] J. Chen and Q. Zhu, "Security as a service for cloud-enabled internet of controlled things under advanced persistent threats: A contract design approach," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 11, pp. 2736–2750, Nov. 2017.

[24] F. Lombardi and R. Di Pietro, "Secure virtualization for cloud computing," *J. Netw. Comput. Appl.*, vol. 34, pp. 1113–1122, Jul. 2011.

[25] J. Arshad, P. Townend, and J. Xu, "Quantification of security for compute intensive workloads in clouds," in *Proc. IEEE 15th Int. Conf. Parallel Distrib. Syst.*, Dec. 2009, pp. 479–486.

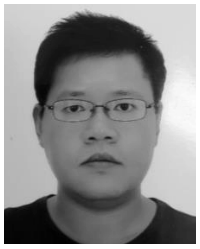
[26] (2007). *CVE Details: The Ultimate Security Vulnerabiltiy Data Source*. [Online]. Available: https://www.cvedetails.com/cve/CVE-2007-4593

[27] O. P. Kreidl, "Analysis of a Markov decision process model for intrusion tolerance," in *Proc. IEEE Int. Conf. Dependable Syst. Netw. Workshops*, Jun./Jul. 2010, pp. 156–161.

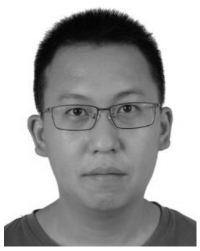
[28] B. B. Madan, K. Gogeva-Popstojanova, K. Vaidyanathan, and K. S. Trivedi, "Modeling and quantification of security attributes of software systems," in *Proc. IEEE Int. Conf. Dependable Syst. Netw.*, Jun. 2002, pp. 505–514.

[29] D. M. Nicol, W. H. Sanders, and K. S. Trivedi, "Model-based evaluation: From dependability to security," *IEEE Trans. Dependable Secure Comput.*, vol. 1, no. 1, pp. 48–65, Jan. 2004.
[30] B. B. Madan, K. Goševa-Popstojanova, K. Vaidyanathan, and K. S. Trivedi, "A method for modeling and quantifying the security attributes of intrusion tolerant systems," *Perform. Eval.*, vol. 56, nos. 1–4, pp. 167–186, Mar. 2004.
[31] A. Årnes, K. Sallhammar, K. Haslum, T. Brekne, M. E. G. Moe, and S. J. Knapskog, "Real-time risk assessment with network sensors and intrusion detection systems," *Comput. Intell. Secur.*, vol. 3802, pp. 388–397, 2005.
[32] V. B. Mendiratta, "Probability and statistics with reliability, queuing and computer science applications," *Interfaces*, vol. 34, no. 5, pp. 407–408, 2004.
[33] J. Lowry, "An initial foray into understanding adversary planning and courses of action," in *Proc. DARPA Inf. Survivability Conf. Expo. II*, Jun. 2001, pp. 123–133.
[34] J. Lowry and K. Theriault, "Experimentation in the IA program," in *Proc. DARPA Inf. Survivability Conf. Expo. II*, Jun. 2001, pp. 134–140.
[35] F. Xu, F. Liu, and H. Jin, "Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud," *IEEE Trans. Comput.*, vol. 65, no. 8, pp. 2470–2483, Aug. 2016.
[36] R. A. Popa *et al.*, "Enabling security in cloud storage SLAs with cloudproof," in *Proc. USENIX Annu. Techn. Conf.*, 2011, pp. 355–368.

HAN XU is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of Electronic Science and Technology of China. His current research interests include cloud computing, and performance modeling and optimization.



XIWEI QIU received the B.S. degree in electronic and information engineering from Jilin University, Changchun, China, in 2004, and the M.S. degree in software engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2009, where he is currently pursuing the Ph.D. degree in computer science with the School of Computer Science and Engineering. His current research interests include cloud computing, reliability modeling and optimization, and energy-efficient computing.



YONGPAN SHENG received the bachelor's degree in network engineering from the Chengdu University of Information Technology in 2011 and the master's degree in software engineering from the University of Electronic Science and Technology of China in 2014, where he is currently pursuing the Ph.D. degree. His research focus is in knowledge graphs and distribution system.

LIANG LUO received the Ph.D. degree from Beihang University, Beijing, China, in 2013. He is currently a Lecturer with the Computer School, University of Electronic Science and Technology of China. His research interests include cloud computing, distributed systems, energy efficiency modeling, and reliability modeling and optimization.

YANPING XIANG received the Ph.D. degree from the National University of Singapore, Singapore, in 2003. She was a Research Fellow with the Industrial and Systems Engineering Department, National University of Singapore. She is currently a Professor with the Collaborative Autonomic Computing Laboratory, University of Electronic Science and Technology of China, Chengdu, China. Her research interests include cloud computing, intelligent agents, and decision making.