Policy-based Security Provisioning and Performance Control in the Cloud

Viswanath Nandina, José Marcio Luna, Edward J. Nava, Christopher C. Lamb, Gregory L. Heileman and Chaouki T. Abdallah

Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, U.S.A.

Keywords: Cloud Computing, Security, Control Theory, Access Control, Usage Management.

Abstract: In this paper we describe the development of a system that provides security provisioning and performance controls over content in a cloud environment. Using an approach grounded in Usage Management and Control theory we are able to successfully provision resources in multiple cloud systems. Providing increased security comes with a cost of reduced performance. Therefore a variable performance control model for different levels of security is proposed. This control model allocates Virtual Machines adaptively so that a desired performance measure lies between predefined upper and lower bounds as agreed in the Service Level Agreement.

1 INTRODUCTION

The bane of the existence of cloud computing has been the inability to ensure security while maintaining performance. Despite the rising popularity surrounding cloud computing, disengaging users from hardware needs has long been criticized for being unable to provide a trustworthy solution to the problem of managing the trade-off between security and performance (Zissis and Lekkas, 2012). Organizations and users have been hesitant to adopt cloud computing because of current cloud systems' inability to provide varying levels of security to various types of data. Currently, many organizations have requirements to assure that data cannot be altered by intentional attacks. Thus some organizations invest in their own computing enterprise systems with rigorous security and reliability protections. There exists a global demand for an automatic control system to enforce security rules so that operations can be directed to private or public cloud systems. Cloud computing continues to be plagued by infrastructural failures and potential security problems (Takabi et al., 2010). Over the past few years major infrastructure providers ranging from Amazon to Microsoft have suffered from extensive and unexpected system downtime. This has led to the development of tools like Amazon's CloudWatch, which can monitor provisioned infrastructure for failures and help secure additional infrastructure when required.

We are pursuing a novel approach for applying Usage Management (UM) (Jamkhedkar et al., 2010) to resources (Virtual Machines and data sets) in a cloud computing environment (Jamkhedkar et al., 2011). We demonstrate how UM can be applied to cloud computing by controlling the resource based on a particular policy set. Likewise others in the area (Takabi and Joshi, 2012) have their own policy management framework. We are incorporating multi-level security on our UM policies. The overhead of the application service provided by the cloud increases as the security requirements become more stringent (*e.g.,* military applications). Therefore, we propose a control theoretic approach to regulate application performance and guarantee that the negotiated SLAs and the security requirements are satisfied.

Though cloud-dependent systems are becoming more stable as a result of these and other efforts, they still have yet to integrate secure provisioning into these kinds of bursting schemes. We present a mechanism that provides robust system and information usage management in tandem with performance monitoring and alerting. This mechanism can securely provision systems on demand based on a unified performance and sensitivity profile. We describe notionally how we expect this system to work, the primary concepts it embodies, and describe our experiences implementing the system using Eucalyptus, Amazon's Elastic Compute Cloud, and Amazon's Simple Storage Service.

In a cloud computing environment an application may have different performance measures, such as,

Nandina V., Luna J., Nava E., Lamb C., Heileman G. and Abdallah C..
Policy-based Security Provisioning and Performance Control in the Cloud.
DOI: 10.5220/0004378705020508
In *Proceedings of the 3rd International Conference on Cloud Computing and Services Science* (CLOSER-2013), pages 502-508
ISBN: 978-989-8565-52-5

response time, throughput or latency. In previous works such as (Nathuji et al., 2010), an approach assuming that the cloud can be modeled as a Multi-Input-Multi-Output (MIMO) system is implemented to regulate performance in the cloud. From this perspective the cloud is seen as a black box with inputs and outputs. The outputs are the different performance measures of interest, while the inputs are values related to the amount of available virtual resources in the cloud, *e.g.,* CPU utilization and available memory in a Virtual Machine (VM), as well as the amount of available VMs in a cluster. The authors assume that the hypervisor allows for a fine-grained control of the parameters of the Virtual Resources in place. However, based on (Vliet and Paganelli, 2011; Lim et al., 2009), in a public cloud service, specifically in Infrastructure-as-a-Service (IaaS) such as EC2 from Amazon, the variation of the internal parameters of the VMs and therefore the response of the system are coarse-grained. This imposes several difficulties to apply methodologies such as Linear Quadratic Regulation as in the CPU problem in (Yao et al., 2010) which is effective to regulate MIMO systems.

In this work, we follow the trend proposed by (Lim et al., 2009) called *proportional thresholding* which is suitable for Single-Input-Single-Output (SISO) systems and overcomes the coarse-granularity of this problem. The main goal is to confine the desired performance to an interval of values rather than regulate it to a specific reference value. However, in order to improve the response of the controller, we have added on-line model identification (Kulhavý, 1987; Haykin, 2002; Hellerstein et al., 2004), which allows the update of the model parameters to overcome the typical workload variations of the cloud.

We implement a policy based usage management system in multiple cloud environments. This system treats resources with specific attributes that may be provisioned according to usage policies. We regulate the performance of running virtual machines to meet the service level agreements applying control theory. This approach is conceptually similar to multi-level security system implemented in a cloud computing environment. We will consider an operational environment that includes a mix of public and private cloud computing resources, each of which can be categorized as having a certain level of trust.

The rest of the paper is as follows: Section 2, describes how usage management goes beyond access control. Section 3 explains the Usage Management system for Cloud computing, as well as the system configuration and the technologies used to build the system. Section 4 deals with performance control and finally Section 5 provides concluding remarks.

2 USAGE MANAGEMENT

A conceptual UM system is described in Figure 1 for a cloud computing environment. The UM system determines if a user can be granted access to a resource, such as a data file. In a cloud computing environment, once a user is granted access to a resource, there is a need to control not only how the resource can be used, but also where. Our primary goal is to develop and implement a secure, robust, and inter-operable attribute-based usage management system for data transactions in cloud computing. This system will merge usage management systems with modern cloud computing technologies. Given a data resource with an associated sensitivity characteristic, one part of the proposed cloud UM process will be to determine on which type of cloud computing system the iresource will be made available.

2.1 Beyond Access Control

Usage management (UM) is defined as the management of the usage of resources (and data) across and within computing environments.(Jamkhedkar et al., 2010) Usage management incorporates characteristics of traditional access control and digital rights management. In order to be effective, it must be flexible enough to be scalable and interoperable enough to provide services across different computational environments. In an implementation of UM, the first action provided by the system is access control. A user logs into a system with credentials and with context that is provided by the user, and/or determined automatically. The next role of the UM is to ensure that data is used in an environment that complies with security policies that are either specified in an associated license, or are applicable to a security category for unclassified data.

For instance, a user might specify at which sensitivity level, or classification level he or she wishes to operate and UM checks the contextual information before providing access to any resources. Based on the examination, UM commands a cloud computing system to instantiate a VM and load an image that contains the necessary security mechanisms. Then it transfers the actual data to the VM. In this paper, we use VM images as the controlled resources, where each image has a set of policies associated with it that describe the circumstances under which that image can be used.

Likewise, each user has an identity that describes their credentials and environment information, also known as context. Usage management occurs when a user's identity information is compared with a resources' policies to determine if that user is qualified to use that particular resource. Once he gets access to the resource the UM system monitors how the resource is being used based upon the policy requirements.

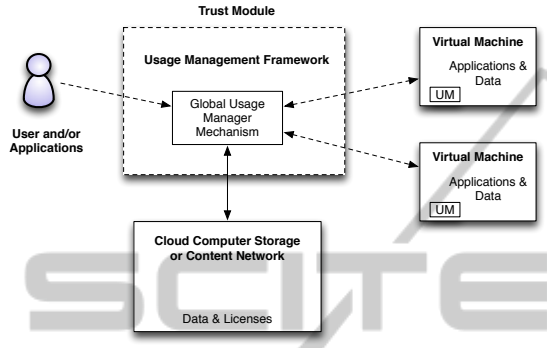

Figure 1: Hierarchical UM operation concept.

3 USAGE MANAGEMENT SYSTEM FOR CLOUD COMPUTING

In order to provide an assured information sharing capability in a cloud computing environment, a UM system capable of interacting with, and controlling, different provider systems is required. The high-level conceptual view is shown in Figure 2. Each system will likely be slightly different, so a common framework, or ontology, is required in order to use common policy semantics so that the policies can be applied consistently in each system. A policy generator will be required in order to generate the licenses that will be needed for each data set, and the policies will use the common ontology as a basis.

Based on the policies that are specified in a license for a particular resource, the UM first grants access to the resource. Then, based on the policies, the UM instantiates a VM in an appropriate cloud system and copies the data to the VM for processing. Once a data set has been transferred to a VM, without additional controls, the user and associated applications can use the data with no restrictions. In order to assure that all resources are used in accordance with the policies, UM capability is also required within the VM for control.

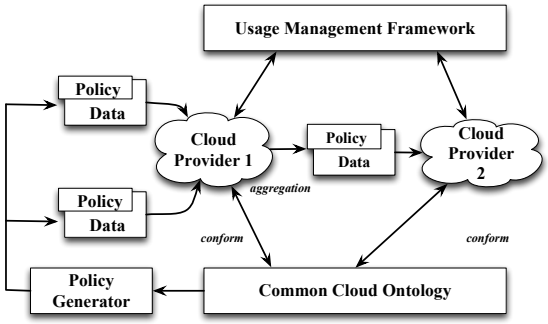

Figure 2: Usage Management in a Multi-Cloud Environment.

3.1 System Architecture/Model

For a complete UM system in a cloud computing environment, a hierarchical design is required. Our proposed system implementation includes the following key elements:

- Usage Management Mechanism: This component acts as a central controller that manages communication between all components of the usage management framework and with external services, such as storage and content networks. The usage management mechanism provides an interface to the user. Different applications can use the common interface provided by the usage management mechanism.
- Trust Module: This module includes a capability to dynamically update the trust values of the cloud resources and update the policies within the licenses. The global Usage Management mechanism in the trust module will then force actions in response to changes, as necessary.
- Virtual Machine Usage Management Mechanism: As mentioned previously, UM within each VM is necessary to ensure that users and applications use data and other resources only in ways that are allowed by the policies specified within the licenses.

3.2 Usage Management within a Virtual Machine

The key elements of a usage management mechanism that will be used in a VM are shown in Figure 3. When we consider UM at the VM level, the control is implemented with a finer level of granularity than is possible when implementing UM with a centralized mechanism because the centralized mechanism cannot control the actions within the individual VMs.

The process of license enforcement within a VM using a variant of our existing UM design takes place in six steps. In step 1, an application queries the usage management mechanism about whether or not a particular action *act* on a given resource by a given user

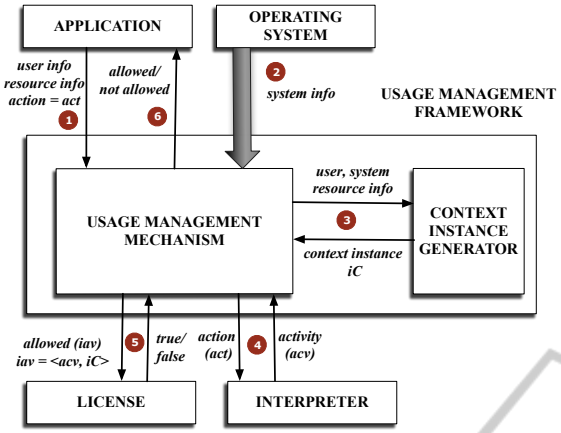

Figure 3: Operation of the Usage Management Framework within a Virtual Machine.

is allowed. As a part of the request, the application provides information about the user, the resource, and the action that is to be performed. In step 2, the usage management mechanism obtains the current state of the computing environment from the operating system. The type of information obtained by the usage management mechanism depends upon the manner in which the context is specified. This information may include current location, day, date, time, IP address, etc. In step 3, the usage management mechanism generates the context instance *iC*, by using the current values of system parameters, user information, and resource information. This is followed by step 4, where the usage management mechanism queries an interpreter to determine the activity *acv* that corresponds to the *act* information provided by the application. Once the activity corresponding to the action is obtained, the usage management mechanism generates the activity instance denoted by the tuple $iav =< acv, iC >$. In step 5 the usage management mechanism invokes the allowed *iav* function provided by the license, the license executes an allowed *iav* function, and returns a Boolean value to the usage management mechanism. Finally, in step 6, the usage management mechanism notifies the application whether or not the action is authorized.

The UM design provides the basis for deciding which operations are allowed and which are not. The outcome of these decisions must be transmitted, in a verifiable manner, to enforcement mechanisms in the operating system executing within the VM. The decision must be verifiable in order to ensure that the decision is properly enforced.

3.2.1 Operational Details

The operational details of our usage management system are visualized in Figure 4. By means of a web

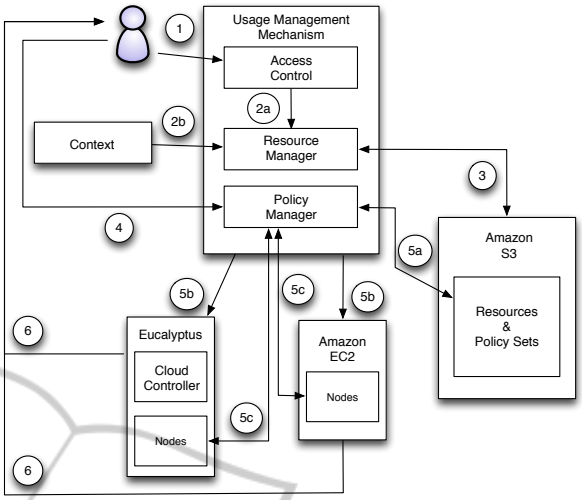

Figure 4: Component diagram of Usage Management for hybrid Cloud-Based System.

browser user interface, the user is presented with a login dialog box. The user enters his/her credentials to log into the system. The credentials and the context information are forwarded to the Resource Manager. Once the user has been authenticated, a list of available resources is presented on the user's browser. This list of resources varies depending on user context. If the user selects a resource, the Policy Manager retrieves the desired resource from a cloud computing repository like Amazon S3. The Policy Manager then instantiates an appropriate Virtual Machine specified to handle resources with the same security context. Depending on the context and policy requirements, resources can be moved from public to a private cloud or vice versa. The Policy Manager also monitors the instance and returns references about the performance of the node to the Policy Manager. Using our control-theoretic approach, we regulate the performance measures of the system that are covered by SLA requirements.

3.2.2 Implementation

The implementation of the system is shown in Figure 5. The application is written using Ruby on Rails and we use MySQL for storing user context. When a user logs in, we use Rails to authenticate the user. Once the user is authenticated we use Resource Manager written in Ruby to decide what resources the user

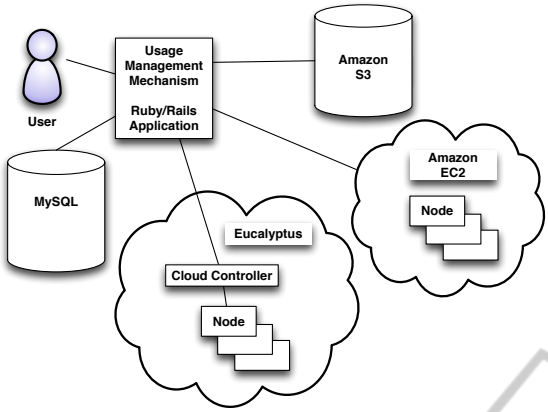

Figure 5: Technology Architecture.

can access by comparing context information with the policy set of the resources and the list of resource references are returned to the user interface. Then the user selects a resource and the policy manager retrieves the resource from Amazon S3 based on the policy information. The policy manager sends the resource to a VM running appropriate image either to Eucalyptus or to Amazon EC2 cloud. It is important to note the actual resources and corresponding policy set are stored in Amazon S3 and some of the features of the Eucalyptus cloud system have been omitted from our diagram for the sake of brevity. With the help of our software we monitor the VMs for performance as defined in the SLA requirements and control the VMs appropriately.

4 PERFORMANCE CONTROL

As mentioned before, different levels of security are considered in the Cloud. Therefore, some virtual resources will be equipped with more complex security infrastructure than others. In Figure 6 we illustrate several VMs, indicated by the hollow circles, that are grouped in clusters depending on the different levels of security. As the demand for security on the available VMs increases, the overhead and therefore the cost of more secure VMs increases as well. That is why our assumption is that there are more VMs with lower security guarantees (Sec. Level K) than the ones that are more secure (Sec. Level 1). One of our main assumptions is that the VMs within each subgroup are identical in resources. Therefore, in order to get an estimate of the model consistent with the available resources we propose to identify a different model for each level of security. In a similar way, we propose to implement a different controller for each level of security as shown in the blocks labeled *Ctrl* in Figure 6.

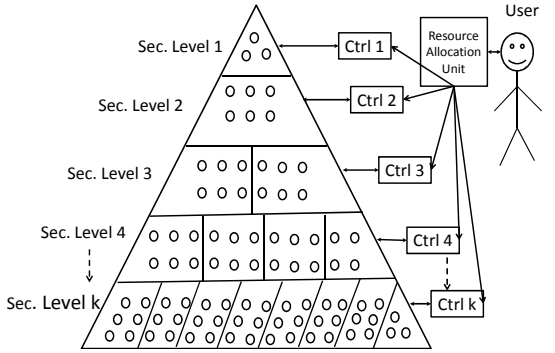

Figure 6: Performance Control of virtual resources for different levels of security. The level of security increases from bottom to top. Notice that the amount of resources decreases as its level of security increases.

The control method we use in our approach relies upon the availability of a mathematical model relating virtual resources and performance measures of the system. Based on (Nathuji et al., 2010), given an input signal u_n and a state signal y_n of the cloud, the general model of the cloud interactions is given by a set of nonlinear difference equations. The existent identification techniques for nonlinear systems are not suitable to be implemented in real-time, furthermore the amount of required learning data for the identification algorithm turns out to be impractical except for the simple cases. In previous works on control of CPU utilization, control of power in data centers and cloud computing described in (Nathuji et al., 2010; Yao et al., 2010; Luna and Abdallah, 2011), a linear time-invariant realization of the system is accurate enough as long as the control inputs of the system are constrained to a relatively ”small“ interval. The control loop including the actual plant, the model identification unit and the controller are shown in Fig. 7.

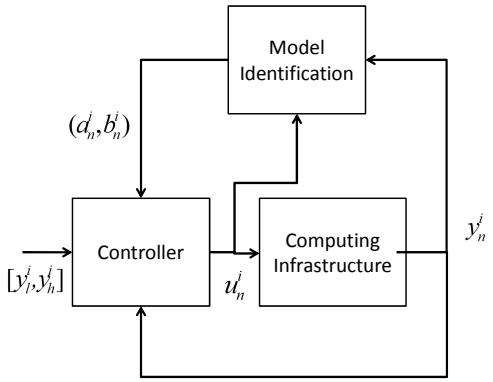

Figure 7: Block diagram of the control loop.

Based on (Hellerstein et al., 2004), our model is approximated by the following difference equations,

$$y_{n+1}^i = a_n^i y_n^i + b_n^i u_n^i, \quad (1)$$

where y_n^i is the state signal representing the performance measure at the n-th discrete time instant, u_n^i is the input signal which represents the number of active VMs in the i-th cluster where $i = 1, 2, \ldots, L$, where L is the total number of clusters of VMs. The system parameters $a^i \in \Re$ and $b^i \in \Re$ are parameters to be estimated by Recursive Least Square (RLS).

4.1 Proportional Thresholding

Following the methodology proposed by (Lim et al., 2009), the desired result of our controller is illustrated in Figure 8. In the plot, the chosen performance measure is the CPU utilization of an entire cluster of VMs. The green dashed lines determine the lower y_l and upper y_h bounds of the desired interval where the performance measure should be confined. The blue vertical lines indicate whether a VM is turned on (+1VM) or turned off (-1VM) in the system. Notice that the performance remains in the interval $[y_l, y_h]$ at all times while the lower bound changes and the upper bound remains fixed.

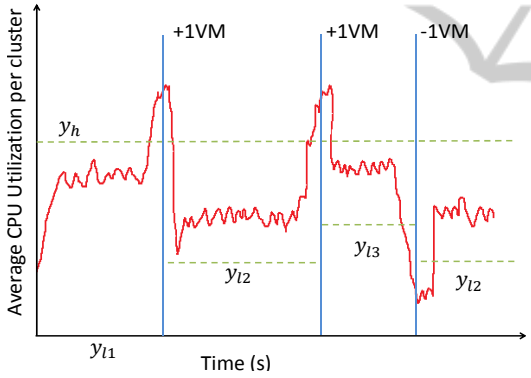

Figure 8: Plot of the desired response. The performance is confined to the interval $[y_l, y_h]$.

The bound that remains fixed y_h is chosen based on the SLA. To determine the appropriate value of the changing bound y_l the authors in (Lim et al., 2009) proceed to take a single VM to obtain an empirical mathematical expression relating the performance measurement of interest (CPU utilization in this example) and the workload. In this case, it is not difficult to expect this expression to be a monotonically increasing function. Once this model has been approximated, it is possible to approximate the total workload of the cluster given that a specific number of VMs are activated. Therefore, depending on the empirical model we can obtain a function $y_l = f(y_h, M)$ with M being the number of currently activated VMs to construct the interval $[y_l, y_h]$. Notice that the fixed bound can be y_l and by a similar procedure $y_h = g(y_l, M)$ can be calculated as well.

Once the empirical model has been determined a modified integral control is proposed as follows,

$$u_{n+1}^i = \begin{cases} u_n^i + K_n^i(y_n^i - y_h) & \text{if } y_h < y_n^i \\ u_n^i + K_n^i(y_n^i - y_l) & \text{if } y_n^i < y_l \\ u_n^i & \text{otherwise} \end{cases} \quad (2)$$

with u_n^i being the number of active VMs in the cluster i and the gain K_n calculated based on the desired transient response of the linear approximation in (1). Different than (Lim et al., 2009) we propose an adaptive gain that changes depending on the values of a_n^i and b_n^i to keep the system closer to the desired transient response. Notice from (2) that if the performance measure gets greater than the upper bound of the interval y_h, then the system activates more VMs. If the performance measure gets less than the lower bound of the interval y_l, then the system deactivates VMs. Otherwise, the current number of activated VMs remains the same. The empirical models that define the relationship between the workload defined by the number of threads and CPU utilization and latency are being currently developed.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we first described the primitives and approaches currently available that we have used to enable simple SLA-centric control over resource utilization and processing based on performance and security sensitive attributes. Thereafter, we described in some detail a system architecture currently realizable with modern open-source tools that enables this kind of dynamic information control. Finally, we discussed our experiences with a prototypical implementation of our proposed system architecture.

We have adopted a control systems approach based on model identification and proportional thresholding. It is capable of guaranteeing performance regulation while satisfying the security requirements of the applications served by clouds. One of the main contribution of the proposed methodology is the addition of control theory to UM policies. It opens a set of possibilities supported by well known computer science tools which are complemented with mathematically supported control theory approaches.

We have multiple clouds because different cloud systems will have different security characteristics. We have demonstrated that we can move data from a cloud repository to an appropriately configured VM within a cloud system based upon the policies specified in the license. We enforce the Policies within

the VM so that the use of those resources is regulated by the policies of the resource. This approach helped us to design and develop a policy based usage management system in conjunction with control theory to regulate performance and provide security to multiple cloud systems.

In the future, we plan to move away from our current web-centric model, examining more data-centric tooling, though we expect to remain committed to open source tools. We will also incorporate more standards, like the Extensible Access Control Markup Language (XACML), to describe policies and controls, and work to establish a clear model behind our work in order to more deeply understand the intrinsic limitations of this problem domain. Currently, we are developing the necessary models, as well as designing the strategies involving the migration of resources to guarantee the stability of the controller.

REFERENCES

Haykin, S. (2002). *Adaptive Filter Theory*. Prentice Hall.

Hellerstein, J., Diao, Y., Parekh, S., and Tilbury, D. M. (2004). *Feedback Control of Computing Systems*. John Wiley & Sons.

Jamkhedkar, P. A., Heileman, G. L., and Lamb, C. C. (2010). An interoperable usage management framework. In *Proceedings of the tenth annual ACM workshop on Digital rights management*, DRM '10, pages 73–88, New York, NY, USA. ACM.

Jamkhedkar, P. A., Lamb, C. C., and Heileman, G. L. (2011). Usage management in cloud computing. In *IEEE CLOUD*, pages 525–532.

Kulhavý, R. (1987). Restricted exponential forgetting in real-time identification. *Automatica*, 25(5):589–600.

Lim, H., Babu, S., Chase, J., and Parekh, S. (2009). Automated control in cloud computing: Challenges and opportunities. In *Proc. of 1st Workshop on Autom. Ctrl for Datacenters & Clouds.*, pages 13–18, Barcelona.

Luna, J. M. and Abdallah, C. T. (2011). Control in computing systems: Part ii. In *IEEE Multi-Conference on Systems and Control*, Denver, CO.

Nathuji, R., Kansal, A., and Ghaffarkhah, A. (2010). Q-clouds: Managing performance interference effects for qos-aware clouds. In *Proceedings of the ACM European Society in Systems Conference 2010*, pages 237–250, Paris, France.

Takabi, H. and Joshi, J. B. D. (2012). Policy management as a service: An approach to manage policy heterogeneity in cloud computing environment. In *HICSS*, pages 5500–5508. IEEE Computer Society.

Takabi, H., Joshi, J. B. D., and Ahn, G.-J. (2010). Security and privacy challenges in cloud computing environments. *IEEE Security and Privacy*, 8(6):24–31.

Vliet, J. V. and Paganelli, F. (2011). *Programming Amazon EC2*. Oŕeilly Media Inc.

Yao, J., Liu, X., Chen, X., Wang, X., and Li, J. (2010). Online decentralized adaptive optimal controller design of cpu utilization for distributed real-time embedded systems. In *Proceedings of the 2010 American Control Conference (ACC'10)*, pages 283–288, Baltimore, MD.

Zissis, D. and Lekkas, D. (2012). Addressing cloud computing security issues. *Future Gener. Comput. Syst.*, 28(3):583–592.