An Approach to Security-SLA in Cloud Computing Environment

Carlos Alberto da Silva
Institute of Computing - Unicamp
Campinas, Brazil
beto@lasca.ic.unicamp.br

Paulo Licio de Geus
Institute of Computing - Unicamp
Campinas, Brazil
paulo@lasca.ic.unicamp.br

***Abstract*—The lack of novel security controls for the cloud might arise from the fact that Cloud Computing is the convergence of many different technological areas, including Utility Computer, Computational Grid, Autonomous Computing, Virtualization and Service Oriented Architectures. These underlying areas have been independently addressed by existing general-purpose security controls, but we noticed that each current cloud security control was mapped to multiple controls from the existing, general-purpose control frameworks. We also noticed a great demand for not only patterns but also specification, monitoring and security management mechanisms for cloud environments. We reason that this scenario might require a different approach, one where the specification of security controls, geared to meet the needs of services users, may be achieved through the use of Security Service Level Agreement—Security-SLA. Security may then be improved by automating the Security-SLA.**

***Keywords*-Security-SLA, Security Management, Security Metrics, Cloud Computing Security**

I. Introduction

Despite cloud services' market growth and massive investments, there is a great concern about security in these environments. According to [1], security issues are considered as the main obstacle against service migration to cloud environments.

The growing concern and dissatisfaction with security in cloud services is the result of a combination of several factors, among which may be mentioned: the lack of knowledge of technical characteristics and risks in cloud environments [2]; the lack of well-defined interoperability standards [3]; loss of control of data and applications [4]; the failures in computing clouds that resulted in unavailability of services, data loss and information leakage [2]; and the lack of guarantees related to the safety level [5].

Driven both by the increasing demand for the use of services in cloud computing, and the large number of security issues in these environments, various institutions started working on typifying services and standards specifications for interoperability and security in cloud computing. Although such actions are a significant step towards the creation of secure environments, most of these efforts are still at a preliminary stage and will require considerable time to get mature and to be adopted, as interests of providers and the pressure from the consumers of services amount.

978-1-4799-7162-6/14/$31.00

The use of levels of security service (Security-SLA) is pointed out as an important tool for the management of security in cloud computing [6] agreements. Despite this, one notices in the literature a lack of concrete work dealing with the specification and monitoring of these agreements. When treated, the solutions describe little about the representation of such agreements and adopt monitoring systems developed for traditional computing architectures.

When further analyzed, those solutions do not look fully prepared to monitor cloud environments because: i) existing tools have little or no support for monitoring SLAs, for agreement representation patterns and for managing agreements on a level basis; ii) the information used for tracking arrangements depend on collection mechanisms that are executed on the machine being monitored itself. In infrastructure cloud services, virtual machines are controlled by the user, which means that the installation of such mechanisms, in addition to depending on the user cooperation, is still subject to incompatibilities caused by differences between operating systems, libraries etc. Furthermore, they are sensitive to tampering in cases where the user is malicious; and iii) existing tools do not consider events that occur in cloud environments, such as creation, suspension, termination and migration of virtual machine execution.

In a bid to foster the adoption of Security-SLA mechanisms, this article presents two contributions: a solution to automate the creation of Security-SLA and a description of a monitoring process for services and devices that make up cloud computing's infrastructure.

II. Theoretical Aspects

In this section we present concepts that will help bring up our contributions.

A. Unmeasurable Qualities

When trying to formalize risk (R), at times there is a need to determine tangible values for intangible assets. Risk is directly linked to the degree of probability of a threat to occur and to the degree of negative impact of the incident caused by the threat to the organization [7], while also measuring the implemented protection effectiveness:

$$R = \frac{(\mathrm{GPO} \times \mathrm{GIN})}{\mathrm{GEP}}$$

Where:

GPO: probability of occurrence of the threat;
GIN: degree of Negative Impact of the incident caused by the threat to business;
GEP: degree of Effectiveness of the implemented protection.

These variables are intangible and unmeasurable. Overall, the qualities specified in an SLA can be classified into measurable and unmeasurable. The measurable qualities are those that can be measured automatically by means of metrics. While the unmeasurable qualities do not allow an automatic measurement, they cannot be evaluated by a method that results in a single value. In these cases, sets of secondary metrics that measure specific aspects of unmeasurable qualities are used.

The following are **measurable qualities** found in IT services [8]:(i) **accuracy:** the error rate threshold for the service during a specific period of time; (ii) **availability:** probability that the service will be available when needed; (iii) **capacity:** number of concurrent requests that the service supports; (iv) **cost:** cost of service; (v) **latency:** the maximum time between the arrival of the request and the time to complete the request; (vi) **provisioning time:** time required for the service to become operational; (vii) **reliability of messages:** guaranteed delivery of messages; (viii) **scalability:** ability to increase the number of operations performed successfully in a time service.

Now we list **unmeasurable qualities** [8]: (i) **interoperability:** ability of intercommunication with other services; (ii) **modifiability:** frequency of changes (interface or implementation) of service; (iii) **security:** ability to resist unauthorized use while providing service to legitimate customers (clients/ tenants).

B. Service Level Agreement

The specification of guarantee parameters assures that the quality of services is an essential mechanism in environments where outsourcing is used. This section discusses the importance of service levels as a way to specify such guarantees, its use on information technology and security services, and finally ways of representing such agreements.

The Service Level Agreement (SLA) is part of the service contract between provider and customer, and describes the desired quality of service (QoS) [9]. An SLA alone does not guarantee that the specified qualities are met, but it defines the necessary monitoring mechanisms, points out the responsibilities and defines punishments and compensations if conditions are not met.

In information technology (IT) services, SLA use takes a different approach than that of telecommunication services. The agreement shall represent both customer's and provider's expectations. As such, obligations may be specified for both parties. The scope of information contained in the agreement is also differentiated.

It is crucial to emphasize that, in the context of an SLA, service level monitoring is as important as their specification. For this purpose, metrics are used to assess compliance with the desired qualities of service. The way these metrics are measured depends on the type of service and quality features that one wants to measure.

C. Security-SLA

The increasing use of outsourced IT services causes a growing concern with issues involving privacy and security [6]. Thus, it is natural for such issues to be addressed in SLAs, which allow the customer to specify security levels that must be guaranteed for the contracted services. However, the specification of SLAs involving security features (Security-SLA) presents challenges that involve the specification of security levels, the representation of these levels and finally monitoring them.

In the literature, the definition of security parameters can be done in two ways: through security policies or from security metrics.

The specification of security settings through policies, as proposed in [10], considers the Security-SLA as a set of policies expressed in standard language (e.g. WS-Policy [11]). Although this approach is able to clearly specify the desired levels of security, the use of policies fail when specifying mechanisms for monitoring, and ignore the representation of various members of SLA information.

This specification from security metrics is a commonly used method that allows definition not only of the security parameters but also of the monitoring process. Unlike the specification of policies, the specification from metrics is based on a set of security metrics that allows checking whether a particular goal (control) is being fulfilled or not.

D. Security Metrics

Although security is an unmeasurable quality [8], [12], it is common to use security metrics to assess the security state of an environment.

Security metrics are tools that provide accurate and current information about the security state of an environment, allowing for an evaluation of operations and security controls in their own environment [13].

The strategies for monitoring cloud computing services using security metrics can be summarized as follows: i) "black-box" external monitoring, where measurements are performed on the host (outside the VMs) and involves the available host OS components, such as firewall, log files, hypervisor and virtual network interfaces; ii) "black-box" internal monitoring, by using similar monitoring mechanisms that applies to each kind of service offering. In the industry these services are referred to as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), respectively. In the worst case (IaaS), only Virtual Machine Introspection (VMI), given by the LibVMI library [14], is available. Introspection allows access to the VM's memory from the hypervisor, enabling information query for data structures kept by the VM's operating system kernel.

We emphasize that all values of security metrics, generated by static and dynamic monitoring, are converted to scale values [0-4], representing the range of security values as [Critical, High, Medium, Normal, Zero] [15].

III. Related Work

Commercial providers offen provide their own solutions for monitoring cloud services. The Amazon AWS platform offers CloudWatch [16], a monitoring system offered as a service for the control of resources and application services. Microsoft Windows Azure Suite[17] has the Azure Fabric Controller, responsible for monitoring and managing servers and coordinating resources for the applications. Google App Engine[18] provides a set of APIs that allow the use of monitoring solutions such as CloudStatus[19].

In Emeakaroha et al. [20], it is presented a monitoring solution called LoM2HiS that is part of an architecture to detect SLA violations. LoM2HiS is composed of a set of agents based on the SNMP protocol, responsible for collecting low-level metrics and sending them to monitors, which then aggregate them into high-level metrics to be used by the SLA threat violation monitor.

There are two main methodologies used in the specification of metrics for SLA-Security. One uses security policies [21] and standards as a starting point for the derivation of metrics. Righi et al. [22] proposed a method for metrics and parameter values validation based on the analysis of measurements made in the service infrastructure.

The other uses the Goal-Question-Metric (GQM) methodology [23], with a process originally proposed for empirical measurements in software testing, based on well-defined goals. The method consists of a measuring model comprised of three levels: i) Conceptual (goal), which is the measurement target; ii) Operating (question), in which the target is refined into a set of operational issues; and iii) Metric, which is a set of metrics that quantitatively answers questions specified at the upper level. In security, this methodology is applied in several studies, such as the GQM model being used together with the COBIT framework [13] to specify metrics for SLA-Security in cloud computing, and the GQM model being used to build a metrics hierarchy to generate an index of security in cloud computing [15].

IV. Methodology

The methodology builds on the concept that the customer, when hiring a service in the cloud (SaaS, PaaS or IaaS), may choose from a portfolio of security metrics that will be continuously monitored by the environment.

Within this approach, a database holds two classes of security metrics, according to their functionality. The infrastructure device class consists of all the hardware devices and related cloud software, such as networking, firewalls, routers, proxies, operating systems etc on which monitoring agents will run to generate security metrics. The service class consists of all SaaS, PaaS or IaaS hardware/software components that provide the service contracted by the customer, with the monitoring agents running on those assets generating security metrics about the Virtual Machine (VM).

A. Automatic Security-SLA

Figure 1 represents the proposed lifecycle of Security-SLA management for cloud computing environments, which is based on the following phases: (1) Definition: this phase is focused on the selection of the infrastructure and service security metrics, its features and the definition of quality parameters that will be provided to customers. A database with all the security metrics (portfolio) is offered; (2) Negotiation: in this phase are defined values for the security metrics parameters (range 0–4), cost to the customer and penalties in case the Security-SLA is violated; (3) Implementation: the security metrics are prepared according to the available infrastructure devices that will allow for the service execution in the environment; (4) Execution: it is the phase when monitoring security metrics for the infrastructure devices and service takes place. Specified quality parameters (SLO) are evaluated for compliance with the Security-SLA; (5) Evaluation: in this phase the provider assesses the security quality provided; (6) Re-negotiation: deals with the service ending, be it for reasons of contract expiration or for Security-SLA re-negotiation.

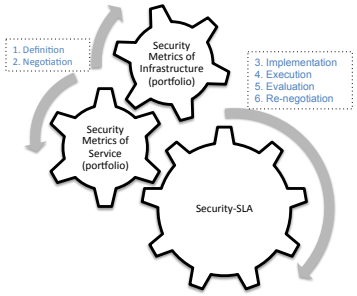


Fig. 1. Lifecycle of a Security-SLA

Figure 2 presents a simplified overview of this Security-SLA management process for cloud computing environments. As today's business systems typically consist of layers of complex systems, user-level Security-SLAs cannot be directly mapped to physical infrastructure. Services can be composed of other more fundamental services, maybe even provided by third parties. Consequently, a gradual mapping of higher-level Security-SLA requirements onto lower levels, and aggregation of lower-level resources to higher-level ones is crucial to allow binding of user-level Security-SLAs to the infrastructure. This vertical flow of information must carefully reflect service interdependencies. In addition to Security-SLAs, vertical information flow also covers monitoring, tracking and accounting data, having to support intermediation and negotiation processes at each layer. The Security-SLA management process may deal with different stakeholders, namely customers, service and infrastructure providers, and also various business steps such as business valuation, contracting and sales. The illustration also shows the role of software providers responsible for creating components with predictable behavior. In this context, one notices the integration of multiple levels, as there are several interested parties (suppliers of software/services/infrastructure

and customers), various roles (IT people, experts, customers), various types of services, various aspects of service level (availability, performance etc), all under the full lifecycle of the Security-SLA (definition, negotiation, implementation, execution, evaluation and re-negotiation).

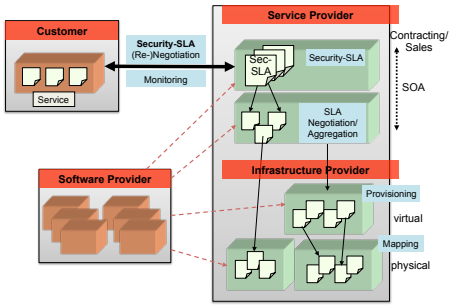


Fig. 2. Security-SLA management in Cloud Computing

B. Monitoring

Monitoring agents are specialized programs responsible for the monitoring process. Each program is tasked with collecting information from existing components in the infrastructure. Such information will be used in the Security-SLA validation.

When an agent runs, it gets the Security-SLA parameters: service that will be monitored, execution parameters and some metric identifiers that will be measured. "Place" specifies the elements where the measurement is done, like VM, Firewall, IDS/IPS etc. "Procedure" specifies whether the type is black-box or not. "Frequency" of measurement in hours. SLO specifies the contract value in the 0–4 range. Finally, "Incidence" specifies the percentage of samples that stayed above the SLO value.

To validate the model, the monitoring process performs two steps:

i) The values collected by security metrics ([0–4]) are classified as: true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN);

ii) Validation indicators are calculated for the model:

a) **Precision:** represented by $P = \frac{TP}{TP+FP}$, indicates the percentage of events correctly classified as incident among those which were classified as incidents;

b) **Recall:** represented by $R = \frac{TP}{TP+FN}$, indicates the percentage of correctly classified incidents among all events that are effectively incidents;

c) **F-measure:** represented by $F = \frac{2 \times P \times R}{P+R}$, is the harmonic mean between precision and recall;

d) **Accuracy:** represented by $A = \frac{TP+TN}{TP+TN+FP+FN}$, indicates the percentage of correctly classified events.

C. Case Study

A case study was developed and tested on a cloud computing environment based on OpenNebula [24], on a machine with a 2.8 GHz Intel i7 Quad core processor and 32GB RAM running Gentoo Linux and the KVM hypervisor. The customer chooses MySQL Enterprise Edition as a SaaS service and as an infrastructure service Intrusion Detection and Prevention System (IDPS). During the negotiation, the client specified that monitoring would be performed 10 times (events in between two samples being acumulat to the next sample). The system under test is responsible for human resources management at an University and holds about 400 tables, 200 users and 5 administrators. Table I describes the security metrics that compose the Security-SLA.

TABLE I
SECURITY METRICS CHOSEN BY THE USER

Item	Description (Metric)	Value of Metric
2	**Infrastructure Cloud Computing**	$Met_2 \geq 3$
2.4	**Intrusion Detect and Prevention System**	$Met_{2.4} \geq 3$
2.4.1	Packet Fragmentation	$Met_{2.4.1} \geq 4$
2.4.2	Stream Segmentation	$Met_{2.4.2} \geq 3$
2.4.3	Remote Procedure Call Fragmentation	$Met_{2.4.3} \geq 3$
2.4.4	Recovery from Abnormal System Shutdown	$Met_{2.4.4} \geq 4$
2.4.5	Security Events Records	$Met_{2.4.5} \geq 4$
2.4.6	Evasion Attacks	$Met_{2.4.6} \geq 3$
9	**SaaS - Cloud Computing**	$Met_9 \geq 3$
9.1	**Database - MySql**	$Met_{9.1} \geq 2$
9.1.1	Default User Service Account	$Met_{9.1.1} \geq 2$
9.1.2	Insecure User Account	$Met_{9.1.2} \geq 2$
9.1.3	Default TCP Port	$Met_{9.1.3} \geq 2$
9.1.4	SQL Injection	$Met_{9.1.4} \geq 2$

1) IDPS Metrics: As an example of the IDPS metric, the following parameters were chosen:

Metric Name: Stream Segmentation

Description: The Stream Segmentation Security Metric ($Met_{2.4.2}$) is monitoring unusual activity on the network, like the remote host advertising a zero window size, dropped TCP connections and session timeouts. By manipulating the way in which a TCP stream is segmented, it is possible to evade detection by some firewalls and IDPS. When doing that, an attacker could overwrite a portion of a previous segment in a stream with new data in a subsequent segment. This method could allow the attacker to hide or obfuscate the attack on the network.

Formula: $Met_{2.4.2} = Count(Incidents)$

SLO Value: 3

Incidence: 90.00%

Table II describes the distribution of sample values (0–4 range) for the Stream Segmentation Metric ($Met_{2.4.2}$), for each monitored incident and their percentage of occurrence.

TABLE II
SAMPLES OF STREAM SEGMENTATION METRIC

$Met_{2.4.2}$	Incidents	Percentage
4	32	0.15%
3	1,505	6.96%
2	15,219	70.37%
1	4,422	20.45%
0	448	2.07%

Based on data from Table II, Figure 3(a) presents the visual result of monitoring the Stream Segmentation Metric ($Met_{2.4.2}$) during the evaluated time span (1 to 10, i.e. the radii in the illustration). The hired SLO value was ≥ 3, the

measured average MA value was 2 in the period, and Incidence total was 7.11% (percentage of incidents of levels 3 and 4: 0.15 + 6.96), however this is in sharp contrast with the contracted value of 90.00%. Therefore, one can conclude that not only there is a problem with the hired security level, but the relation between the measured 7.11% and the expected 90.00% values for the Incidence also suggests that the delivered security is very poor.

2) MySQL Metrics: **Heuristics**. We implemented two heuristic algorithms to compute the incidents over the MySQL service:

a. **Log:** Uses black-box external monitoring, analyzes log files of the database, and identifies and records incidents. Let n be the number of records in the log file, m the number of operations and l the number of permissions. The asymptotic complexity is $O(n \log ml)$
b. **Interface:** Uses black-box internal monitoring, runs a PHP code inside the system interface and each command in the interface, verifies and records the incidents. Let n be the number of commands in the interface, m the number of operations and l the number of permissions. Its asymptotic complexity is $O(nml)$

As an example of MySQL metric, the following parameters were chosen:

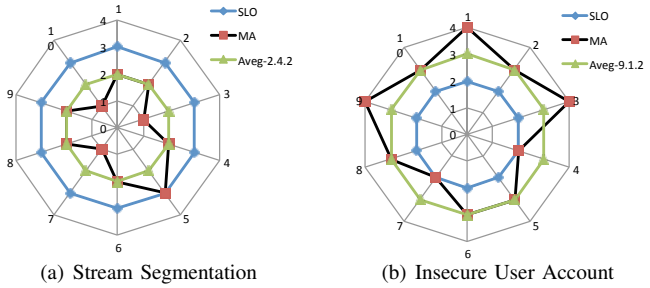
Metric Name: Insecure User Account Security
Description: The Insecure User Account Security Metric ($Met_{9.1.2}$) is monitoring whether the customer used a default user account instead of an administrator account. The transaction log for the database is checked for the combination: source (Administrator or User), type of operation (select, update, drop, alter and create), and permission of the operation.
Formula: $Met_{9.1.2} = Count(Incidents)$
SLO Value: 2
Incidence: 80.00%

Table III describes the distribution of sample values (0–4 range) for the Insecure User Account Metric ($Met_{9.1.2}$), for each monitored incident and their percentage of occurrence.

TABLE III
SAMPLES OF INSECURE USER ACCOUNT METRIC

$Met_{9.1.2}$	Incidents	Percentage
4	5,678	19.84%
3	18,104	63.25%
2	3,624	12.66%
1	1,000	3.49%
0	215	0.75%

Based on data from Table III, Figure 3(b) presents the visual result of monitoring the Insecure User Account Metric ($Met_{9.1.2}$), during the evaluated time span (1 to 10, i.e. the radii in the illustration). The hired SLO value was ≥ 2, the measured average MA value was 3 in the period, and Incidence total was 95.75% (percentage of incidents of levels 2 to 4: 19.84 + 63.25 + 12.66), yielding a value well above the contracted 80.00%. Therefore, one can conclude that security



(a) Stream Segmentation (b) Insecure User Account

Fig. 3. Behavior of the Security Metrics

as contracted has not only been met, but in fact the relative position of the 95.75% figure towards the hardest target of 100% suggests that security was quite good.

V. CONCLUSIONS AND FUTURE WORK

We presented a substantial contribution to make an automatic way of contracting a Security-SLA using as basis a portfolio of security metrics for the infrastructure and services classes. We also introduced a new model to view information about security through a range of values (0–4) and treated the problem of managing intangible and unmeasurable numbers. Moreover, we proposed a new way of managing security levels (top-down view) that considers values for each security metric with its respective risk, Quality of Service (QoS) and impact. Separating Security-SLA in two reference security value classes allows for an abstracted visualization of security and helps to easily spot which security items present values below the expected values. Thus, the customer may have a more tangible feeling of how the hired service is being protected.

As future work, we consider the development of dynamic mappings between security metrics to automatically identify services running on the VM whose type of service was not specified. This would be based on monitoring security anomalies and matching them against performance signatures, and new, non-invasive techniques to monitor and compute the risk and impact over the environment.

ACKNOWLEDGMENT

The authors would like to thank CAPES and Fundect (Process #23/200.308/2009) for his financial support.

REFERENCES

[1] Ian Foster and Yong Zhao and Ioan Raicu and Shiyong Lu, Cloud Computing and Grid Computing 360-Degree Compared, Grid Computing Environments Workshop (GCE '08), pages 1-10, 2008.
[2] Shashikala P. Subashini and Veeraruna R. Kavitha, A survey on security issues in service delivery models of cloud computing, Journal of Network and Computer Applications, pages 1-11, 2011.
[3] Christina N. Hoefer and Georgios Karagiannis, Taxonomy of cloud computing services, Proceedings of the 4th IEEE Workshop on Enabling the Future Service-Oriented Internet (EFSOI'10), pages 1345–1350, 2010.
[4] Ronald L. Krutz and Russell Dean Vines, Cloud Security: A Comprehensive Guide to Secure Cloud Computing, pages 384, John Wiley & Sons, Inc., 2010.
[5] ENISA, Cloud Computing: Benefits, risks and recommendations for information security, European Network and Information Security, 2009.

[6] Martin Gilje Jaatun and Karin Bernsmed and Astrid Undheim, Security SLAs - An Idea Whose Time Has Come?, pages 123-130, Lecture Notes in Computer Science, Springer Berlin/Heidelberg, 2012.

[7] NIST Cloud Computing Security Reference Architecture, NIST Special Publication 500-299, is available at collaborate.nist.gov/twiki-cloud-computing/pub/Cloud Computing/CloudSecurity/NIST_security_Reference_Architecture _2013.05.15_v1.0.pdf. Accessed in July 05, 2014.

[8] Philip Bianco and Grace A. Lewis and Paulo Merson, Service Level Agreements in Service-Oriented Architecture Environments, Carnegie Mellon University (SEI), 2008, available in http://www.sei.cmu.edu/reports/08tn021.pdf. Accessed April 09, 2014.

[9] Diana Berberova and Boyan Bontchev, Design of Service Level Agreements for Software Services, Proceedings of the International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing (CompSysTech '09), ACM, 2009.

[10] Valentina Casola and Antonino Mazzeo and Nicola Mazzocca and Massimiliano Rak, A SLA evaluation methodology in Service Oriented Architectures, Quality of Protection, pages 119-130, Volume 23, Springer US, 2006.

[11] W3C, Web Services Policy 1.5 - Framework, World Wide Web Consortium, Available in http://www.w3.org/TR/ws-policy/. Accessed in July 05, 2014.

[12] Leanid Krautsevich and Fabio Martinelli and Artsiom Yautsiukhin, Formal approach to security metrics: what does more secure mean for you?, Proceedings of the Fourth European Conference on Software Architecture: Companion Volume, pages 162–169, ACM, 2010.

[13] Nia Ramadianti Putri and Medard Charles Mganga, Enhancing Information Security in Cloud Computing Services using SLA Based Metrics, School of Computing - Blekinge Institute of Technology, 2011.

[14] VMITools. Virtual machine introspection tools. Available in https://code.google.com/p/vmitools/. Accessed in July 05, 2014.

[15] Carlos Alberto da Silva, Anderson Soares Ferreira, and Paulo Licio de Geus. A metho- dology for management of cloud computing using security criteria. In Proceedings of the IEEE Latin American Conference on Cloud Computing and Communications, LatinCloud12, Porto Alegre, Brazil, November 2012.

[16] Amazon Web Services, Inc. Amazon CloudWatch. Available in http://aws.amazon.com/cloudwatch. Accessed in July 05, 2014.

[17] Microsoft Corporation. Microsoft Windows Azure. Available in http://www.windowsazure.com. Accessed in July 05, 2014.

[18] Google, Inc. Google App Engine. Available in https://developers.google.com/appengine/. Accessed in July 05, 2014.

[19] Hyperic. CloudStatus. Available in http://www. hyperic.com/products/cloud-status-monitoring. Accessed in July 05, 2014.

[20] Vincent C. Emeakaroha, Ivona Brandic, Michael Maurer, and Schahram Dustdar. Low level metrics to high level slas - lom2his framework: Bridging the gap between monitored metrics and sla parameters in cloud environments. In International Conference on High Performance Computing and Simulation, 2010, HPCS10, pages 4854. IEEE Computer Society, 2010.

[21] Ronda R. Henning, Security Service Level Agreements: Quantifiable Security for the Enterprise?, Proceedings of the 1999 Workshop on New Security Paradigms (NSPW '99), pages 54–60, ACM, 1999.

[22] Rafael R. Righi and Felipe R. Pellissari and Carlos B. Westphall, Sec-SLA: Specification and validation of Metrics for Security Oriented Service Level Agreements, IV Workshop in Computing Systems Security, SBC, Porto Alegre-RS, Brazil, 2004.

[23] Victor R. Basili and Gianluigi Caldiera and H. Dieter Rombach, The goal question metric approach, Encyclopedia of software engineering, Volume 2, pages 528–532, 1994.

[24] OpenNebula. OpenNebula Project. Available in http://www.opennebula.org/. Accessed in July 05, 2014.