Security SLA based monitoring in clouds

Nesrine Kaaniche, Mohamed Mohamed, Maryline Laurent, Heiko Ludwig

Security SLA based Monitoring in Clouds

Nesrine Kaaniche[1], Mohamed Mohamed[2], Maryline Laurent[1], Heiko Ludwig[2]
[1]SAMOVAR, Telecom SudParis, CNRS, University Paris-Saclay
[2]IBM Research, Almaden Research Center, San Jose, CA, USA

***Abstract*—Nowadays, Cloud providers revise the terms of their Service Level Agreements (SLAs) to include security provisions due to their criticality for their customers. In order to speed up their adoption by service providers and consumers and to make them more actionable, security SLAs monitoring should be described in a machine-processable, agile and extensible way. Several tools for SLA management are available on the market but most deal with performance metrics and do not refer to security properties. There are other tools for monitoring cloud security, in a non-SLA way. However, they are not associated with SLA management systems. We propose an extension to an SLA language (i.e., rSLA) to enable the description of security requirements in an SLA document. We also extend the rSLA framework by a security methodology that makes use of known tools and that enables continuously checking that the security requirements are respected during runtime according to the SLA document.**

***Keywords*– Security Level Agreement, Security Monitoring, Cloud Security Monitoring, SLA**

I. Introduction

Cloud computing has presented novel ways to continuously build and deploy applications at scale to heterogeneous environments (private and public cloud, on premise, hybrid environments). Different providers might have various offerings for similar services. Furthermore, most of the providers give their customers the ability to set up applications in a short time. These applications might be a composition of micro-services, which are often dynamic and change frequently.

The large number of services and the possibility of composing them into a single application makes it challenging to manage them efficiently and control their Quality of Service (QoS). Since these services might have heterogeneous interfaces and might come from different providers any commitment related to their QoS relies on the different Service Level Agreements (SLAs) established with each service provider.

SLAs are widely used to describe and manage the QoS agreed upon between customers and providers. For instance, monitoring SLAs for simple metrics like workload, performance or availability, offers both to the providers and the consumers the needed information to implement mechanisms to prevent or recover from eventual agreement violations [1], [2].

However, there are only few SLA management systems which address cloud environments. Also, security management is less developed than managing operational performance in these environments. Security obligations associated with a service should be explicitly specified in an SLA (in security SLA or what we call Sec-SLA). Moreover, the specification of security SLAs should be backed-up with the right monitoring and enforcement mechanisms that address cloud environments' dynamic nature.

The absence of security management in currently used SLAs, combined with the lack of methods for making objective comparisons between different service offerings, makes it impossible for providers to offer trustworthy services to their customers [3]. As far as we know, almost all existing work in the state of the art does not offer a holistic method to describe security SLAs and to manage them during runtime [2], [4], [5].

In this paper, we propose to extend an existing SLA description language (i.e., rSLA [6]) to support the description of security requirements. We also extend rSLA framework to provide the needed mechanisms that allow the automatic setup of the needed facilities to manage the security SLA and enforce them according to their specification. Our extension enables using the most suitable monitoring tools to enforce the security requirements. The aforementioned extension called sec-rSLA gives stakeholders an overview of the security level of their service, without requiring detailed manual analysis of log files. The overall proposed system, dynamically inspects the SLA document and put in place the needed security monitoring tools to collect the data, evaluate them against the objectives and eventually execute enforcement and reporting actions.

Unlike existing solutions, which either do not take into account security metrics in the SLA description or do not cover their life-cycle management and enforcement, our proposal enables a holistic approach that starts from the rSLA language concepts allowing the description of security metrics and their aggregation, and goes beyond that to cover the setup of needed facilities to manage the life-cycle of the sec-SLA as well as their enforcement.

The remainder of this work is organized as follows. Section II gives an overview of tools and prototypes that are so far available for security SLA monitoring in clouds. Then, Section IV introduces our approach and Section V gives an evaluation use-case. Finally, conclusions are given in Section VI.

II. Related Work & Problem Statement

Monitoring processes in Clouds serve to collect, process and report information on metrics such as performance or other custom metrics against well defined behavior rules or policies [2], [4], [5]. The main feature of monitoring systems consists in running a mechanism for the detection of the state of the system and performing automatic actions or generating reports for resolutions in a transparent manner on any infrastructure. Several areas in the Cloud practice monitoring such as performance, billing and service availability. However, Cloud security monitoring is considered essential because security concerns are reported as the biggest hurdle for the adoption of the Cloud by enterprises, mainly due to the loss of control. Thus, cloud security monitoring mechanisms will make it possible to implement a relationship of trust and transparency between the cloud service provider and his customers.

A. Security Monitoring in Clouds

Security monitoring in cloud is a dynamic repeated process which engages effective and proactive management of cloud components to identify and respond to threats and vulnerabilities [12]. Security monitoring involves various techniques and methods that are designed to collect event logs from multiple systems and provide comprehensible and valuable data reports that have to be correlated and analyzed. On one hand, these security mechanisms, such as intrusion detection and prevention systems, provide logging information in order to enhance trust, transparency, visibility and get rid of uncertainty surrounding cloud services due to their abstractness.
On the other hand, security monitoring systems involve processes for collecting and analyzing data for events of interest to automatically respond to security-related incidents. Thus, cloud security monitoring has to provide real-time publication of security occurrences to involved actors to efficiently decide their mitigation policies.
As such, from clients' point of view, cloud security monitoring is crucial for enhancing accountability and mutual trust, while (i) helping clients to transparently detect SLA violations and (ii) making cloud services accountable by helping in gathering of evidences to validate security claims of CSPs.
From a cloud provider's point of view, security monitoring constitutes an efficient means to capture current state of cloud resources, detect security breaches and control customers' activities to check malicious use of resources.

We summarize in Table I a collection of tools and prototypes that are available for security SLA monitoring in clouds. We distinguish open-source monitoring tools, proprietary monitoring systems and research prototypes for cloud security monitoring. We notice that most of the reviewed schemes are often limited to intrusion detection in virtual infrastructures.

In [13], the authors state that there is no deployed sec-SLA and point out the related issues. That is, the definition of a security quantifiable metric is challenging due to the fact that security is a collection of properties, varying from service performances to process maturity [14]. Several research activities focus on the quantification of security, measuring it and providing security levels. However, most of these security metrics are deterministic in the sense that a number value was enough to quantify the security level of a system. However, in the last few years this notion has changed dramatically because both researchers and practitioners have found that uncertainty plays a central role in security evaluations.

In 2012, the European Network and Information Security Agency (ENISA) classified security parameters for a security monitoring framework [13]. Beyond parameters definition, methods for measuring parameters in practice were provided. However, security indicators were not defined (*a security indicator is an observable characteristic that correlates with a desired security property, the set of feasible indicator values is expected to form a nominal scale*). Recently, the CUMULUS project[1] provides an attribute-based security vocabulary, which relies on the CSA Cloud Controls Matrix (CCM) [2].

B. Challenges to Security Monitoring in Clouds

While the definition of a cloud SLA monitoring system in a cloud context is difficult due to the increasing dynamism of application environments and the heterogeneity of service interfaces and instrumentation tools, the design of a cloud security SLA monitoring system is much more challenging, considering that:
(1) frequent reconfigurations require the security monitoring system to automatically adapt security modifications;
(2) monitoring policies might not be comprehensively defined to support the detection of violations;
(3) several monitoring functions, such as event detection systems, involve log analysis. Many challenges arise when defining the retention policy of logs, as well as specifying accesses to these sensitive tracing information;
(4) the outsourced service is made up of a set of subsystems that are involved in multiple interactions, and the security level of the monitored service depends on the security of each subsystem and its robustness against threats, threat coverage, and cost;
(5) the security level of each subsystem depends on the security of other subsystems linked to it; and
(6) the overall security level of a service is an aggregated evaluation of a set of security and privacy properties that are interleaved. For example, among non-available data causes, we report the violation of data replication policy, and among the causes of data disclosure, we proclaim the storing data in clear. Additionally, even though several security and privacy properties have been addressed, we still need a careful consideration of data security requirements. That is, these mechanisms are limited to the detection of data leakage. They do not refer to root causes, thus

[1] http://www.cumulus-project.eu
[2] https://cloudsecurityalliance.org/group/cloud-controls-matrix/

TABLE I
CLOUD SECURITY MONITORING TOOLS

ACRONYM	DESCRIPTION
Open-Source Cloud Security Monitoring Tools	
FBCrypt [1]	prevents information leakage by encrypting the I/O between a client and a user VM using a Virtual Machine Monitor (VMM).
Snorby [2]	application for network/host security monitoring with includes Intrusion Detection Systems (Snort IDS).
Commercial Cloud Security Monitoring Tools	
CipherCloud [3]	cloud security platform that provides a set of protection controls including risk assessment, data protection (searchable encryption), tokenization, data loss prevention, key management and malware detection and extensive user activity and anomaly monitoring services.
CloudFlare [4]	improves web services and mobile applications' availability and protects websites (eg; re-route traffic, limit abusive bots and crawlers.
CloudPassage [5]	provides cloud security as a service. It aims to secure virtual servers in cloud infrastructures and provides File Integrity Monitoring (FIM), while administering firewall automation, vulnerability monitoring, network access control, security event alerting, and assessment. It also provides security applications such as access management, software vulnerability scanning and log-based IDS (Splunk IDS).
MARS [6]	a Cisco monitoring system. It is designed to monitor logs and threats.
Research Prototypes for Cloud Security Monitoring	
CASViD [7]	a cloud application for detecting SLA violation at the application layer. It includes tools for resource allocation, scheduling, and deployment.
QoS-MONaaS [8]	it allows to describe in a formal SLA the key performance indicators of interest and the alerts in case of SLA violation.
HAIL [9]	is a distributed cryptographic system, to prove data retrievability. HAIL, *High Availability and Integrity Layer*, differs from remote checking frameworks. In fact, it considers a distributed setting in which a client must spread a file across multiple servers with redundancy and only stores a small constant state locally.
CloudProof [10]	CloudProof is a secure cloud storage system [10]. In CloudProof, customers can not only detect violations of integrity and freshness, they can also prove the occurrence of these violations to a third party. This proof-based system is critical to enabling security guarantees in SLAs, wherein clients pay for a desired level of security and are assured they will receive a certain compensation in the event of cloud misbehavior.
CloudSec [11]	it provides active, transparent and real-time security monitoring for multiple concurrent VMs hosted on a cloud platform in an IaaS setting.

[1] https://github.com/danigrant/fbcrypt
[2] https://github.com/snorby/snorby
[3] http://www.ciphercloud.com
[4] https://www.cloud are.com
[5] https://www.cloudpassage.com
[6] http://www.cisco.com/c/en/us/products/security/security-monitoring-analysis-response-system

allowing to have a comprehensive monitoring report, and to ensure greater transparency for consumers.

III. TOWARDS SECURITY MONITORING IN THE CLOUD

When applying SLA monitoring concepts to security, the complexity grows easily, as there is a *semantic gap* between the cloud client, which generally has specific security requirements, but often does not have the expertise on security taxonomy, and the cloud provider which wants to express the security level of its services with respect to detailed metrics.

Hence, along with the challenges raised in section II, we propose an efficient approach for collecting forensic security evidence in the cloud environment making it more accountable and transparent for clients. Our approach is based on the following five steps, namely: *formal representation*, *evidence translation*, *security enforcement*, *privacy preserving evidence management* and *distributed trails*.

- *Formal representation* – the first step consists in extracting from each system which elements can serve as digital tracing information or evidence. Such representation permits to define the set of evidence requirements for each layer of the security monitoring system.
- *Evidence translation* – it consists in translating the evidence requirements into concrete elements of the operating-system and target-applications' programming interfaces. This step permits to inform the cloud runtime which specific parameters and functions that have to be audited as security metrics.
- *Security enforcement* – the third step involves putting in place the needed mechanisms that dynamically activate auditing functions to collect digital evidence related to the metrics defined in the second step when needed. Indeed, using the action mechanism offered by rSLA, we can describe the actions to be executed if any violation occurred. An action is an abstract concept in rSLA that we can specify for each use case. It basically refers to a target endpoint that will execute the real action. We can define an action for each security vulnerability or an action for multiple vulnerabilities. For instance, a simple action is the reporting that consists in notifying the responsible parties whenever a violation occurred.
- *Privacy preserving evidence management* – the goal is to preserve users' privacy, from the collection of audit records to their analysis. That is, only authorized functions are able to collect audit records and access to data reports. For this purpose, this step relies on the use of attribute based mechanisms [15]. Indeed, authorized functions are able to collect encrypted

audit records, such as only entities having the set of required credentials can decrypt enciphered data. In addition, access to audit reports can be performed anonymously [16], such as it ensures unlinkability between the different sessions while preserving the anonymity of the requesting entity.

- *Distributed trails* – taking advantage of the computation and storage capabilities of the different cloud nodes, each node has to provide audit reports. Then, an authorized aggregating node is responsible for performing operations on received records in a privacy-aware manner, while preserving the authenticity of the resulting audit report.

As far as we know, most of existent security monitoring systems rely on interfaces that either introduce too much overhead or do not have visibility at the abstraction levels to collect data related to the security metrics. Hence, based on the proposed steps in our approach, cloud providers enable their clients to select a level of security. This assumes a price model which relies on dynamic proactive mechanisms and depends on collected forensic evidences as well as enhanced privacy protection. Thus, if an application is compromised, the chance of being able to reconstruct the intruder's actions increases as more detailed evidences is available. Security levels are means for maintaining confidence between customers and cloud providers. This has led several states to enter into international agreements to recognize the validity of certification and, at the same time, to develop national assessment systems in order to maintain control over System Information Security (SIS) product evaluations. This is the case of France, which, like 24 other countries (Germany, South Korea, United States, $\cdots$), has signed the mutual recognition agreement according to common criteria. This agreement allows the signing countries to recognize the validity of a *certificate* issued according to these criteria by a certification authority (chosen by one of these countries).

In France, the National Agency for Information Systems Security (ANSSI) approves the CESTI (Information Technology Security Evaluation Center) to issue a certificate according to common criteria with regards to the defined security levels. The certificate must indicate the level of EAL (Evaluation Assurance Level) certification obtained by the product, with the EAL1 rating being the lowest and the EAL7 rating being the highest.

As such, our approach considers a set of security auditing mechanisms for each security level. Indeed, once a security level is selected, the client requires auditing reports through proactively collected evidences, corresponding to this specific level.

In the following, we present our security SLA monitoring approach, in accordance with the aforementioned steps, based on the rSLA language and framework [6].

IV. Overview of the sec-rSLA Approach

The main objective of our cloud security SLA system is to provide a security description in SLAs, which is both flexible enough to support the dynamism of applications and efficient enough to report and enforce the overall security level of the outsourced service. In the

A. Language Extension

```
sla do
  tenant "ExampleClient"
  provider "ExampleProvider"
end
basemetric do
    name "data_integrity"
    type "event_set integer"

    measurementdirective do
      entity "/data"
      type "event_set integer"
      source "http://xlet.com/data/integrity"
    end
    security do
      vulnerability
          "{"name":"vul1","name":"vul2"}"
      level "{1,3}"
      availability "high"
    end
    schedule do
      frequency "1"
      unit "m"
      method "every"
    end
end
basemetric do
    name "data_replication"
    type "event_set integer"
    measurementdirective do
      entity "/data"
      type "event_set integer"
      source "http://xlet.com/data/replication"
    end
    security do
      vulnerability
          "{"name":"vul1","name":"vul2"}"
      level "{1,3}"
      availability "high"
    end
    schedule do
      frequency "10"
      unit "s"
      method "every"
    end
end
 compositemetric do
      name "data_alteration_prevention"
      expression do
        method "data_integrity +
            data_replication >= 4"
        select "this_month"
        end
      securityExpression do
      vulnerability
          "{"name":"vul1","name":"vul2"}"
      level "{1,3}"
      availability "high"
      end
  end

  slo do
   name "data_alteration_slo"
   precondition "data_alteration_prevention Not
       true"
   schedule do
```

```
61      frequency "1"
62      unit "m"
63      method "every"
64     end
65   end
66
67   action do
68    condition
          "{"violation":"data_alteration_slo"}"
69    target "http://rXlet.com/vulnerability"
70    parameters "vulnerability,evaluation"
71   end
```

Listing 1. Sec-rSLA sample

Our security SLA monitoring approach consists of the extension of the rSLA system [6], in order to support security metrics monitoring, referred to as sec-rSLA. We extend the rSLA language, such that we added the *security* tag to rSLA language to support different security requirements such as confidentiality, integrity and authentication. Each security requirement corresponds to a set of auditing procedures, executed on the basis of base and composite metrics, with respect to the two first steps introduced in section III, namely *formal representation* and *evidence translation* phases.

Listing 1 illustrates sample statements for the creation of an SLA that contains two base metrics (Lines 5-43) and a composite metric (Lines 44-55) with security requirements (Lines 50-54). We refer to the composite metric in the SLO (Lines 57-65). For instance, the precondition of the SLO (Line 59) says that the precondition is that the composite metric *data_alteration_prevention* is violated (i.e., Not True). If the SLO is violated, an action should be executed (Lines 67-71).

The composite metric *data_alteration_prevention*, verifies if the resource correctly implements security measures to preserve data integrity. Thus, this composite metric aggregates values of the two base metrics: *data_integrity* and *data_replication*. The *data_integrity* base metric checks if the resource correctly stores data, while *data_replication* verifies if the resource correctly deploys the replication policy.

The SLO (*data_alteration_slo*) has a precondition related to the *data_alteration_prevention* composite metric. The precondition says that the SLO is valid whenever the evaluation of the composite metric is true. As described above, this means that the security measures are correctly deployed. Otherwise, if the composite metric is not true, the SLO is supposed violated. In this case, there should be an enforcement action. The listing 1 shows that whenever the SLO is violated, an action will be executed by sending the context of the violation comprising the vulnerability and the evaluation result.

Over and above, we extended rSLA runtime to enable the processing of the newly added security requirements. Whenever the SLA document is agreed upon by the customer and the provider, it is sent to rSLA runtime. This latter will interpret the different parts of the agreement and automatically bind to the offered resources that allow to manage the SLA. These resources are described in the following section.

B. System Model

As introduced in section IV-A, the sec-rSLA gives stakeholders an overview of the security level of the service, without requiring any detailed manual analysis of log files.

Security monitoring is performed by mean of light-weight components called Xlets. Xlets are dynamically bound adapters that abstract the heterogeneity of service management interfaces to rSLA, both for the reading of metrics as well as the acting on the result of SLA evaluation. In our extension, we added new Xlets implementing different algorithms for security monitoring and verification. The extended runtime is able to interpret the SLA document and automatically choose the right Xlets to be used to monitor the security aspects.

In our approach, we consider that a service (S) is presented by a set of elements (storage nodes, data centers, network infrastructure, $\cdots$), that are involved in multiple interactions. Among these elements, we distinguish those that are potential target of cyber-attacks. These elements are referred to as *Vulnerable Elements* (VE), such as:

$$(S) = \{VE_1, VE_2, \cdots, VE_N\}, \tag{1}$$

where N is the number of Vulnerable Elements that compose the service. We note that a VE may represent any element of the stack-layers of the cloud system. Each VE presents the resource that has to be evaluated with respect to a set of *security properties* (SPs). These SPs are based on measurable metrics with collected data from the corresponding algorithm, selected by the Xlets.

Results on SPs' measurements have to be aggregated with respect to dependencies' relationships. These security relationships can be adjusted to any use case in accordance to its security requirements, to express the overall security level of the outsourced service. This is done in our extension to rSLA through the *securityExpression* element added to the *compositemetrics*.

The security level belongs to a *Security Class* (SC). Note that (SC) presents the security level objective of the service (S).
The assessment of the overall security level of the outsourced service is provided by the rSLA evaluation service which is designed to read metrics in homogeneous way. For instance, rSLA service gets this homogeneous data following DMTF standard outputted from the Xlets. These latter play the role of abstracting the heterogeneity of different provider interfaces used to measure security metrics by transforming the data into the standard format that rSLA can handle.

C. Sec-rSLA Model

Figure1 presents how the current security level of a service (S), is evaluated from the instrumentation, based on the different security relationships of VEs.

Recall that when expressing the rSLA document, the author of this document has to describe what can be read

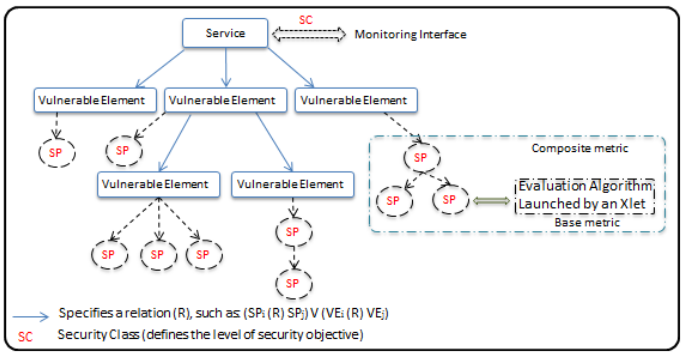

Fig. 1. Sec-rSLA Concepts Overview

by instrumentation into a high-level metric on which the customer wants to have commitment. This metric will be transformed automatically by rSLA service into a binding to an Xlet responsible of the data collection.

A monitored resource (VE) is represented by a couple of sets, such as $VE = \{\mathcal{P}, \mathcal{D}\}$. $\mathcal{P}$ represents the set of security properties that have to be assessed by an evaluation algorithm, such as $\mathcal{P} = \{SP_1, SP_2, \cdots, SP_m\}$, where m is the number of security properties. $\mathcal{D}$ represents the set of dependencies' relationships, such that $\mathcal{D} = \{R_1, R_2, \cdots, R_l\}$, where l is the number of relations of dependency. We distinguish two cases:

- *case A*: $(\{m = 1\} \wedge \{l = 0\})$ – the set $\mathcal{P}$ contains one single security property SP_1, that has to be evaluated. This SP_1 presents a measurable base metric.
- *case B*: $(\{m \neq 1\} \wedge \{l \neq 0\})$ – the set $\mathcal{P}$ includes m security properties related by l relations of dependency. This case corresponds to a composite metric that aggregates several base metrics. Each $\{SP_i\}_{i \in \{1,m\}}$ presents a base metric, while the aggregation relation, specified by the rSLA evaluation service, is a function derived from l relations of dependency.

Our high-level metric is a composite metric and it corresponds to the security class which identifies the security level of the monitored service. This composite metric presents the aggregation of the results on composite metrics, referred to each monitored resource (VE). This gives us our main sets of concepts:

The *base metric* is the elementary security measurement, provided by sec-rSLA. As presented in [6], a base metric has a *measurement directive* and a *schedule*. While a measurement directive specifies how a specific metric can be obtained through an Xlet, the schedule defines the frequency at which measurements are taken.

The *composite metric* aggregates results of base metrics and other composite metrics. The aggregation expressions are described using the relations of dependencies between the different security properties, which we have evaluated as base metrics. While the phrasing of the expression must be type-compatible with the type of inputs, we basically rely on two measure types, namely the boolean result to express if the resource correctly deploys the security algorithm and the integer result to express the level of security (e.g., security level of the enciphering keys) or even the percentage of achievement of such a property (e.g., the number of disclosed fragments from overall data fragments). We note that a boolean input may be transformed to an integer variable so that the security requirements are perfectly adapted to rSLA language.

As presented above, the *Service Level Objective* corresponds to the security level of the monitored service. It serves as a commitment from the service provider to its client. The evaluation of an SLA is the evaluation of each SLO. However, the failure to respect an SLO may induce reconfiguration actions or penalties depending on the agreement. Security SLOs are eventually the ones having more importance and higher penalties. They might lead to a deal cancellation and change of provider.

V. Case Study

To evaluate the efficiency of our sec-rSLA approach, we consider a real world use case. In the following, we first present the different entities involved in the defined scenario (section V-A). Then, we introduce the security requirements and the corresponding monitoring tools in section V-B and section V-C, respectively.

A. Description

For our scenario, we consider the case of an enterprise that stores, processes and manages private financial information for different investors. The enterprise is based on an hybrid cloud infrastructure, relying on different virtual servers, with respect to three layers. The first layer includes servers storing and processing trade transactions, clients' accounts and financial information. These data are then shared with stackeholders. The second layer consists of administrators' servers to ensure basic administration functions, such as pricing service, while the third layer includes virtual servers that are configured to ensure authorized access and use of resources.

B. Formal Presentation

The *formal representation* phase consists on defining the set of requirements for each layer of the enterprise infrastructure. For this use case, we first abstract each layer (S) as a set of vulnerable elements (VE). Hence, we define 4 vulnerable elements, such as: $VE_1 = "API"(ApplicationProgrammingInterface)$, $VE_2 = "CC"(CommunicationChannel)$, $VE_3 = "DC"(DataCenter)$, $VE_4 = "VM"(VirtualMachine)$.

We then define the main vulnerabilities, related to the selected (VE). These vulnerabilities are used to define the set of security properties $\mathcal{P}$ (cf. Table II). They cover the access control of users, data availability, privacy preserving, data storage and data leakage. Table II summarizes the security requirements for different layers, categorized according to data access, data storage and data processing requirements.

TABLE II
MAJOR SECURITY AND PRIVACY REQUIREMENTS

Security Requirements	Details
Data Access Security Requirements	
Access Control	• a fine-grained data access policy has to be enforced to prevent unauthorized access to clients' related data. • it is crucial for the system to be able to revoke certain users and disable their possible use in the future, when needed (e.g; compromised user).
Authentication	• each user has to authenticate within an API, in order to prevent injection of corrupted inputs as well as identities' usurpation. The authentication procedure may be ensured in an anonymous manner to preserve clients' privacy against curious service providers [16]. • to prevent clients from abusing their privileges, it is necessary that a trusted third party should be able to remove the anonymity of users, when needed.
Data Storage and Processing Security Requirements	
Confidentiality	• data must be protected when outsourced to remote servers. • considering that outsourced data have to be protected, the cloud storage provider has to be able to prove to the user that it is encrypting his data at rest, when the provider holds the corresponding encryption keys. • the geographical location of data storage and processing in clouds matters, mainly due to compliance rules and privacy laws may require that sensitive data must be stored and processed under the same jurisdiction. • it is important to combine confidentiality, with respect to the remote storage sever, with search functionality and enable the cloud provider to search encrypted data, based on a *trapdoor* data provided by the client. • when outsourcing encrypted contents in remote servers, it is important to provide the cloud with capabilities to ensure computations over enciphered data.
Integrity	• outsourced data have to be protected against modifications by unauthorized users. • each user must be able to periodically check the integrity of his outsourced data.
Availability	• outsourced data have to be available. As such, each user must be able to periodically check the availability of his outsourced data, with respect to the replication policy, as specified in the SLA contract.

As shown in Table II, the set of security properties is presented by $\mathcal{P} = \{SP_1, SP_2, SP_3, SP_4, SP_5\} = \{$ *access control, authentication, confidentiality, integrity, availability* $\}$.

C. Security Enforcement

The *security enforcement* phase is built upon the *evidence translation* phase that permits to inform the cloud runtime which specific parameters and functions have to be audited as security metrics.
As such, this dynamic auditing phase defines the needed mechanisms that dynamically activate auditing functions to collect digital evidence related to the security metrics when needed. For this purpose, the *security enforcement* phase identifies monitoring tools and specify algorithms that have the essential functions for monitoring one or a set of the properties pointed out in Table II, with respect to the set of dependencies' relations $\mathcal{D}$.

For ease of presentation, we only consider the integrity requirement, following the Listing example presented in section IV-A, with respect to the vulnerable element VE_3. For the integrity security requirement, we identify two kinds of relations of dependency. In fact, the set of dependencies' relations $\mathcal{D}$ is represented by $\mathcal{D} = \{R_1, R_2\}$, such that $R_1 = [SP_4(R)SP_2] \wedge [VE_3(R)\{VE_1, VE_2, VE_4\}]$ and $R_2 = [SP_4(R)SP_5] \wedge [VE_3(R)\{VE_1, VE_2, VE_4\}]$.
Afterwards, relying on $\mathcal{D}$, monitoring tools have to be selected and categorized with respect to their efficiency. For instance, the first relation R_1 deals with dynamically verifying the insurance of data replication policy. There exist different mechanisms and auditing processing permitting to check the existence of multiple copies of outsourced data, as claimed in the SLA contract. For example, Juels and Oprea [17] present *Iris*, a new cryptographic technique that secures outsourced data by ensuring a range of protections from integrity and freshness verification to high data availability. First, Iris encompasses a proof of data retrievability procedure that supports fully dynamic changes. Second, it permits to detect data losses, while auditing drive-failure resilience, based on the RAFT protocol introduced in [18]. In addition, Wang proposes an Identity-based Distributed proof of data possession scheme (ID-DPDP) for multi-cloud environments [19]. The proposed mechanism enables SLA violation detection with respect to data integrity, while supporting private auditing, delegated auditing as well as public auditing under the data

owner authorization.

Finally, we state that each one of these cryptographic mechanisms is deployed as an Xlet consumed by defined base metrics and aggregated through composite metrics. These mechanisms could be easily used in SLO evaluations. Hence, we have to note that there exist some monitoring tools supporting several security requirements that we can consume through the Xlets and present their data in a standard format that rSLA service can process.

D. Summary

In summary, we must note that in order to provide efficient sec-rSLA monitoring, the enterprise has to guarantee a baseline security level (i.e., minimum sec-rSLA) through the system. As such, we recall that sec-rSLA contains security controls that are associated to SLOs. Although security auditing mechanisms rely on standardized frameworks, SLOs and the related security metrics are still defined and managed by the cloud service offered by the enterprise.

Finally, we state that, while the final step of our approach *distributed trails* permits the cloud provider to favor the use of its resources, the *privacy preserving evidence management* step is considered as a supplementary feature which is not subject of audit operations with respect to clients' security needs. It rather permits privacy-aware storage and processing of monitoring reports. Nevertheless, this privacy-capability may be audited by an authorized third party.

VI. Conclusion

In this paper, we present an efficient SLA monitoring system for security properties in Clouds based on the rSLA framework. Our approach simplifies security monitoring and provides a granular way of describing security metrics and composing them into higher levels by aggregating or filtering. Since rSLA is a DSL based on the Ruby language, it tends to be easy to be picked up administrators who generally prefer system-level scripting tools (similar to Chef[3] or Puppet[4]). Moreover, our solution covers the complete life-cycle of the security SLA from its specification to its deployment and management. It also comes with the possibility of executing enforcement actions or reporting. The loose coupling between the different components involved in the SLA management enables the horizontal scalability of our framework. In particular, since most of the components are deployed in Bluemix, we can automatically scale them as needed. Moreover, using Xlets reduces the heterogeneity of the environment and makes it easier to observe and analyze security metrics. Using the proposed security extensions, the benefits of a dynamically executable SLA language and its execution framework can be applied to security guarantees, which are of utmost importance for enterprise use of Cloud services for critical applications.

[3]https://www.chef.io

[4]https://puppet.com

References

[1] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing," *Communications Surveys Tutorials, IEEE*, vol. PP, no. 99, pp. 1 –17, 2012.

[2] K. Alhamazani, R. Ranjan, K. Mitra, F. Rabhi, P. P. Jayaraman, S. U. Khan, A. Guabtni, and V. Bhatnagar, "An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art," *Computing*, vol. 97, no. 4, pp. 357–377, 2015.

[3] K. Bernsmed, M. Jaatun, P. Meland, and A. Undheim, "Security slas for federated cloud services," in *Sixth International Conference on Availability, Reliability and Security (ARES)*, Aug 2011, pp. 202–209.

[4] J. S. Ward and A. Barker, "Observing the clouds: a survey and taxonomy of cloud monitoring," *Journal of Cloud Computing*, vol. 3, no. 1, p. 24, 2014.

[5] G. Aceto, A. Botta, W. De Donato, and A. Pescapè, "Survey cloud monitoring: A survey," *Comput. Netw.*, vol. 57, no. 9, pp. 2093–2115, Jun. 2013.

[6] H. Ludwig, K. Stamou, M. Mohamed, N. Mandagere, B. Langston, G. Alatorre, H. Nakamura, O. Anya, and A. Keller, "rsla: Monitoring slas in dynamic service environments," in *International Conference on Service Oriented Computing (ICSOC)*, November 2015.

[7] V. C. Emeakaroha, T. C. Ferreto, M. A. S. Netto, I. Brandic, and C. A. F. De Rose, "Casvid: Application level monitoring for sla violation detection in clouds," in *Proceedings of the 2012 IEEE 36th Annual Computer Software and Applications Conference*, ser. COMPSAC '12. Washington, DC, USA: IEEE Computer Society, 2012.

[8] O. Adinolfi, R. Cristaldi, L. Coppolino, and L. Romano, "Qos-monaas: A portable architecture for qos monitoring in the cloud," in *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems (SITIS)*, Nov 2012, pp. 527–532.

[9] K. Bowers, A. Juels, and A. Oprea, "Hail: A high-availability and integrity layer for cloud storage," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, ser. CCS '09, New York, NY, USA, 2009.

[10] R. A. Popa, J. R. Lorch, D. Molnar, H. J. Wang, and L. Zhuang, "Enabling security in cloud storage slas with cloudproof," in *Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIXATC'11. Berkeley, CA, USA: USENIX Association, 2011.

[11] A. Ibrahim, J. Hamlyn-Harris, J. Grundy, and M. Almorsy, "Cloudsec: A security monitoring appliance for virtual machines in the iaas cloud model," in *2011 5th International Conference on Network and System Security (NSS)*, Sept 2011, pp. 113–120.

[12] U. M. Ismail, S. Islam, M. Ouedraogo, and E. Weippl, "A framework for security transparency in cloud computing," *Future Internet*, vol. 8, no. 1, 2016.

[13] D. Petcu and C. Crăciun, "Towards a security sla-based cloud monitoring service," pp. 598–603, 2014.

[14] S. de Chaves, C. Westphall, and F. Lamin, "Sla perspective in security management for cloud computing," in *Sixth International Conference on Networking and Services (ICNS)*, March 2010, pp. 212–217.

[15] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography–PKC 2011*. Springer, 2011, pp. 53–70.

[16] N. Kaaniche and M. Laurent, "Attribute-based signatures for supporting anonymous certification," in *European Symposium on Research in Computer Security*. Springer, 2016, pp. 279–300.

[17] A. Juels and A. Oprea, "New approaches to security and availability for cloud data," *Communications of the ACM*, vol. 56, no. 2, pp. 64–73, 2013.

[18] D. Bowers, M. Van Dijk, A. Juels, A. Oprea, and R. L. Rivest, "How to tell if your cloud files are vulnerable to drive crashes," in *Proceedings of the 18th ACM conference on Computer and communications security*, ser. CCS '11, New York, NY, USA, 2011.

[19] H. Wang, "Identity-based distributed provable data possession in multicloud storage," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 328–340, 2015.