Benchmarking Cloud Security Level Agreements Using Quantitative Policy Trees

Jesus Luna, Robert Langenberg, Neeraj Suri
Dept. of CS, TU Darmstadt, Germany
{jluna, robert, suri}@deeds.informatik.tu-darmstadt.de

ABSTRACT

While the many economic and technological advantages of Cloud computing are apparent, the migration of key sector applications onto it has been limited, in part, due to the lack of *security assurance* on the Cloud Service Provider (CSP). However, the recent efforts on specification of security statements in Service Level Agreements, also known as "Security Level Agreements" or SecLAs is a positive development. While a consistent notion of Cloud SecLAs is still developing, already some major CSPs are creating and storing their advocated SecLAs in publicly available repositories e.g., the Cloud Security Alliance's "Security, Trust & Assurance Registry" (CSA STAR). While several academic and industrial efforts are developing the methods to build and specify Cloud SecLAs, very few works deal with the techniques to *quantitatively reason* about SecLAs in order to provide security assurance. This paper proposes a method to benchmark – both quantitatively and qualitatively – the Cloud SecLAs of one or more CSPs with respect to a user-defined requirement, also in the form of a SecLA. The contributed security benchmark methodology rests on the notion of *Quantitative Policy Trees* (QPT), a data structure that we propose to represent and systematically reason about SecLAs. In this paper we perform the initial validation of the contributed methodology with respect to another state of the art proposal, which in turn was empirically validated using the SecLAs stored on the CSA STAR repository. Finally, our research also contributes with QUANTS-as-a-Service (QUANTSaaS), a system that implements the proposed Cloud SecLA benchmark methodology.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics

General Terms

Algorithms, Measurement, Security

Keywords

Cloud security, Security Benchmarks, Security Level Agreements, Security Metrics, Security Quantification

1. INTRODUCTION & CONTRIBUTIONS

With security as a major driver, numerous efforts are ongoing to "create the mechanisms" required to deploy trustworthy Cloud ecosystems with mostly incipient efforts to measure the achieved trust levels. Driving this is the security-related concern in the Cloud user community of *security assurance*, which as mentioned by Schneier [31] is essentially "... demonstrate that your system is secure, because I'm just not going to believe you otherwise". Despite the pervasive nature of Cloud technologies and their advocated economic/technological advantages, applications migration is still limited, in part, due precisely to the lack of this *security assurance* by the Cloud Service Provider (CSP). This lack of assurance, along with the current paucity of techniques to quantify security, often results in users being unable to assess the security of the CSP they are paying for. Despite the assumption that a given Cloud provider "seems" secure, is it actually "secure enough" for my applications? Is my personal data more secure today than before? How do I compare against other providers with regards to security?

Security assurance in Cloud computing has recently taken some initial and promising steps. The Cloud community e.g., workgroups at the European Network and Information Security Agency (ENISA) [12] and the Cloud Security Alliance (CSA) [10], has identified that specifying security in Service Level Agreements (termed as "Security Level Agreements" or SecLA over this paper) is useful to model and assess the security being offered by a CSP. At the same time, the state of the art/practice predominantly focusses on the methodologies to build and represent these Cloud SecLAs, but there is a conspicuous gap on the techniques to *quantitatively reason* about Cloud SecLAs in order to provide security assurance. Specifically the needs are for quantification, aggregation, benchmark, negotiation, prediction and tuning of Cloud SecLAs.

Targeting this core aspect of quantitatively reasoning about Cloud SecLAs, this paper proposes a methodology to *benchmark* the Cloud SecLAs of one or more CSPs with respect to a user-defined requirement, also in the form of a SecLA. In order to automatically and quantitatively reason about SecLAs, our research proposes the use of *Quantitative Policy Trees* (QPT) a novel data structure that can be used to "map", and systematically process these Cloud SecLAs.

Through the use of QPTs our benchmark methodology improves existing state of the art approaches by *(i)* providing fine-grained information about security benchmarks for the Cloud and, *(ii)* providing either quantitative or qualitative SecLA benchmarks depending on the end-user of this information (e.g., either a human decision-maker or an automated Cloud resource broker).

In this paper we perform the initial validation of the contributed methodology by comparing the obtained results with another related approach, which in turn was empirically validated in a previous research paper [24] through a real-world case study, based on the Cloud SecLAs found on the CSA's "Security, Trust & Assurance Registry" (STAR repository [10]).

As a final contribution we also introduce *QUANTS-as-a-Service* (QUANTSaaS), a system that implements the proposed benchmark methodology. QUANTSaaS will be publicly available[1] as a tool to empower Cloud users through providing choices of CSP via the benchmarking of Cloud SecLAs and providing security *assurance*. Further empirical validation of the presented methodology will take place as future work, utilizing the feedback expected from the QUANTSaaS users.

The paper is organized as follows: Section 2 presents the basic concepts behind the paper, Section 3 introduces the proposed benchmark methodology, Section 4 presents and discusses the obtained results, Section 5 surveys related work and finally Section 6 discusses our main conclusions.

2. BASIC CONCEPTS

This section introduces the notions of Cloud Security Level Agreements (cf. Section 2.1) and Quantitative Policy Trees (cf. Section 2.2), that underlie the proposed benchmarking methodology in Section 3. We review the basic concepts of the Reference Evaluation Methodology (REM) [7], a state of the art technique that was applied by Luna [24] to quantify Cloud SecLAs. In Section 4 we will use these basic REM notions in order to compare it with respect to the benchmarking methodology proposed in this paper.

2.1 Cloud Security Level Agreements (SecLAs)

The concept of SecLAs currently exists in varied dimensions (cf. Section 5) and the Cloud is not an exception. The use of Cloud SecLAs has the potential to provide tangible benefits to CSPs especially associated with improved security administration and management practices, thus allowing for transparency to end users. The notion of SecLAs forces a stakeholder to explicitly address security. The end users can also benefit from SecLAs by understanding the costs and benefits associated with this new service model. On one hand SecLAs aim to provide service-based assurance, but on the other hand it is clear that SecLAs are not intended to replace assurance mechanisms for security policy enforcement [15].

The importance of Cloud SecLAs has also been recognized by ENISA: the development of template contracts and service level agreements (SLA) is being highlighted as one of the areas to be addressed in the European Cloud computing strategy. In a recent survey [12] ENISA highlights that many Cloud customers often do not monitor security aspects of their contracted SLA on a continuous basis. This implies that customers are left unaware about many important security aspects related to their services. The risk is that they find out about failing security measures only following a security breach. The survey data shows that while SLAs are often used, and availability is often addressed in these SLAs, security parameters are less well covered.

As shown in Figure 1 (introduced by Bernsmed [3]), a Cloud SecLA usually models the CSP security at the *service level* and, is based on either a set of expert-driven security requirements (e.g., for compliance reasons) or some kind of preliminary threat analysis. The result is a collection of security statements (also called "security provisions" as in Figure 1) in the form *{security attribute, value}* (e.g., *{Backup Frequency, Daily}* and *{Encryption Key Size, 512 bits}*), as also proposed in different industrial and academic works [8], [29] and [23]. In order to be manageable, these security provisions must be organized into "categories" derived from a taxonomy e.g., Savola [30] or the Cloud Security Alliance's Cloud Controls Matrix [9]. This set of security provisions – now organized into taxonomic categories – will finally become a Cloud SecLA "template", which can be used by CSPs to define their own SecLAs. Cloud SecLAs are usually stored in publicly available – and trusted – repositories like e.g., the Cloud Security Alliance's (CSA) "Security, Trust & Assurance Registry" (STAR [10]). Apart from the challenges related with the creation of SecLAs in real Cloud deployments, the current paucity of techniques to *quantitatively reason* about them has proven to be part of the obstacles in using SecLAs, just as mentioned by Almorsy [1] and Luna [23], [24]. To start bridging this gap, this paper contributes a methodology to benchmark Cloud SecLAs.

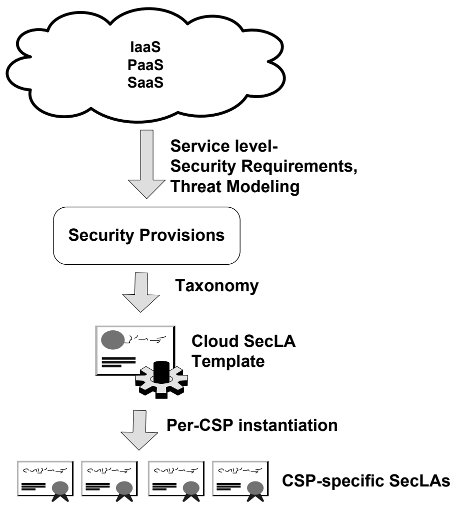


Figure 1: Workflow used to create Cloud Security Level Agreements (SecLAs).

2.2 Quantitative Policy Trees (QPTs)

Despite the advantages of Cloud SecLAs (cf. Section 2.1), usually these documents are informally formatted (e.g., free form text spreadsheets) thus limiting both humans and computers to quantitatively and automatically reason about them. For this reason we propose the "Quantitative Policy Trees", an extended version of classical "AND-OR" trees, to represent Cloud SecLAs an enable the use of quantitative methodologies such as the benchmarking proposed in this paper.

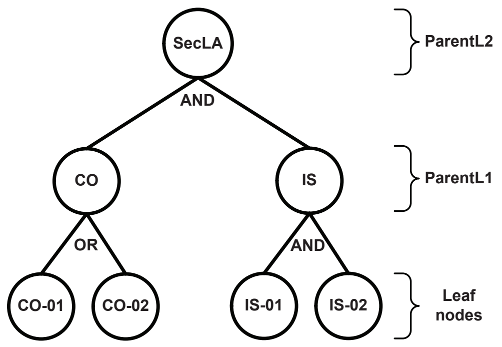


Figure 2: A simple Quantitative Policy Tree (QPT).

As shown in Figure 2, an AND-OR tree is a hierarchical data

[1] `http://quant-security.org`

structure where branches are created with any of these two relationships in order to model the different strategies needed to achieve certain "goal", usually represented by the root node. These are widely used for threat modeling (i.e., "attack trees") and also to model security requirements as proposed by Bistarelli [5]. We utilize AND-OR trees in the following ways:

1. They allow the creation of "patterns" (i.e., Cloud SecLA templates in our case), that can be re-used by other CSPs to design their own SecLAs, therefore taking advantage of the knowledge from the experts that originally created them.

2. AND-OR trees also allow integrating of security requirements and the associated quantifiers.

We elaborate further on QPTs in Section 3.

2.3 The Reference Evaluation Methodology (REM) at Glance

In its basic form, REM *(a)* considers a set of security provisions to evaluate, *(b)* formalizes it to facilitate subsequent evaluation over an homogeneous metric space, *(c)* uses a set of reference levels (known as *Local Security Levels* or *LSL*) to apply a distance criterion, and *(d)* finally obtains a number (also called *Global Security Level* or *GSL*) that corresponds to the policy's security level.

For the purposes of this paper, some basic REM-related concepts are presented and the interested readers are referred to [7] for further details.

The first concept is the formal representation of any security policy by a $n \times m$ matrix, whose n rows represent single security provisions with a maximum of m possible LSLs. For example, if the LSL associated to a Cloud Storage Provider's provision called "File System Encryption" is 3, then the corresponding vector[2] will be *(1,1,1,0)*. The REM concept of LSL is congruent with the notion of security ranges, as presented in [17].

The second REM-concept is a criteria used to quantify GSLs, and defined as the Euclidean distance[3] among whatever pair of REM-matrices *(A,B)*. Just as in Equation 1, the GSL is defined as the square root of the matrix trace of $\left((A-B)(A-B)^T\right)$, where $(A-B)^T$ is the conjugate transpose.

$$GSL(A,B) = d(A,B) = \sqrt{Tr\left((A-B)(A-B)^T\right)} \qquad (1)$$

As a stand-alone security evaluation methodology, REM does not provide any additional metric to quantitatively reason about the LSL and GSL associated with each Cloud SecLA. To bridge this gap, the research presented in [24] developed a set of metrics to perform the security assessment of a CSP based on the REM-quantification of its SecLA.

3. PROPOSED BENCHMARK METHODOLOGY FOR CLOUD SECLAS

The benchmark methodology presented in this section is illustrated in Figure 3, where a predefined *Cloud SecLA template* (e.g., the CAIQ [9] template) and a set of *CSP SecLAs* (e.g., the ones stored in the STAR [10] repository), are the starting point to compute a security benchmark with respect to a user-defined SecLA requirement. The overall intent is to *(a)* systematically quantify the security level associated with each SecLA and *(b)* use the data from *(a)* to guide end-users in choosing the CSP that is closer to their security requirements.

[2]The LSL is usually known as the *L1-Norm* [33] of this vector
[3]Also known as the Frobenius-Norm [32]

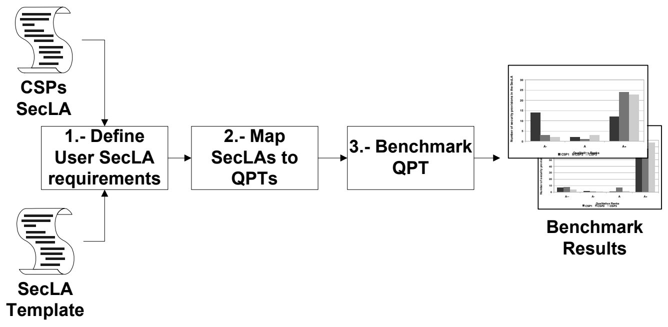


Figure 3: Stages of the proposed Benchmark Methodology.

Considering the basic concepts of SecLA and QPT from Section 2, the following subsections detail each of the progressive blocks of our benchmark methodology.

3.1 Stage 1: Defining User's SecLA requirements

During this first stage, the end-user creates their set of security requirements based on the *SecLA Template* given as one of the inputs in our methodology (cf. leftmost side of Figure 3). This is a key consideration, as in practice security benchmarks can not be performed using SecLAs based on different templates (e.g., ISO27001 and HIPAA). In practice, these *SecLA Templates* are created by multi-disciplinary working groups (e.g., the Cloud Security Alliance's SLA and PLA work groups [11]). In these groups, usually industry and academia design the SecLAs' content (i.e., the security and privacy requirements along with their associated metrics) and discuss their technical, operational and legal issues. Our ongoing collaboration with the CSA aims to align the methodology proposed in this paper, with the real-world SecLAs being developed by their work groups.

User-defined requirements will termed as *User SecLA* in this paper. A *User SecLA* is a distinctive element of Cloud SecLA, where all the security provisions are *weighted* in order to represent their relative importance from the user's perspective (e.g., for some users "Encryption Key Size" might be more important than "Backup Frequency"). The basic guidelines for setting weights to individual security provisions are:

- Each security provision on the *User SecLA* will be associated with a quantitative weight ω_i ($0 \le \omega_i \le 1$).

- The sum of all the weights ω_i associated with a set of sibling security provisions (i.e., those having the same parent category) must be equal to 1.

- The user can choose only specific categories/security provisions to benchmark by assigning $\omega_i = 0$ to those not of interest, as shown in Figure 5 and discussed in Section 4.

Using the previous guidelines and based on our initial empirical validation (cf., Section 4), we have found that a convenient approach for assigning weights is to first set each sibling security provision with the same weight. Second, to identify important provisions (e.g., the ones related with mandatory requirements) and finally, to "transfer" them most of the weight (at least 50%) from those that are not critical for the operation of the Cloud service. Further validation of our methodology (cf. Section 6) will provide us with more real-world experience for setting these weights.

To complete the customization of *User SecLA*, the user can also select the appropriate AND/OR relationships between the different categories of the *SecLA Template*. As inferred from their name, AND relationships will model *hard-requirements* where "Categories A, B and C are *all* required due to regulatory compliance", whereas OR relationships are more adequate to model *soft-requirements* e.g., "Either A, B or C are needed for my to achieve my security goals".

The output of this stage will be three different classes of SecLAs, namely one *SecLA Template*, one *User SecLA* and, one or more *CSP SecLAs*.

3.2 Stage 2: Mapping SecLAs to QPTs

In this second stage, the Stage 1 SecLAs are *transformed* into corresponding QPTs, as introduced in Section 2.2. In order to map a Cloud SecLA to a QPT, we propose the process outlined in Figure 4. This resembles the creation of a SecLA (cf. Figure 1) but in *reverse* order: we start with a set of SecLA's categories and then, via an iterative refinement process, proceed to extract each individual security provision.

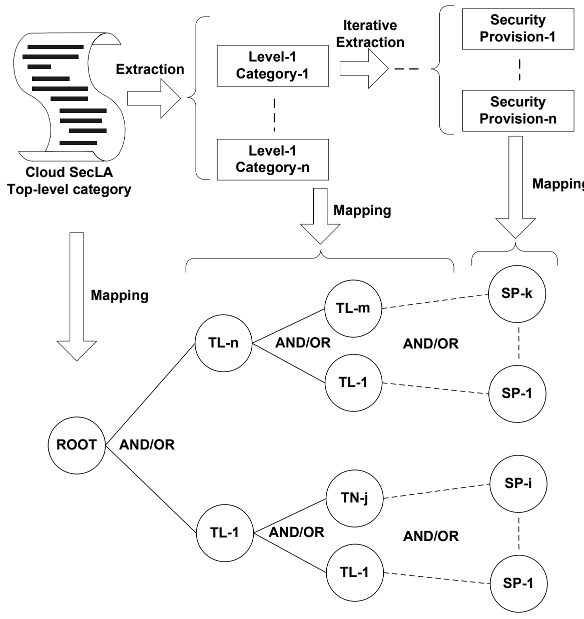


Figure 4: SecLA-to-QPT: mapping a Cloud SecLA into a QPT

Taking into account the inherent tree-like structure of a Cloud SecLA, we propose the following SecLA-to-QPT process:

1. The top-level SecLA category is directly mapped to the QPT's root node.

2. Intermediate SecLA categories (non-leaf nodes), are mapped to the QPT's intermediate nodes taking into account *(i)* their hierarchy in the SecLA (as defined by the *SecLA Template*) and, *(ii)* the AND/OR relationship that exists among these categories (inherited from the *User SecLA*).

3. Finally, individual security provisions (i.e., the duple *{security attribute, value}* introduced in Section 2.1) are mapped to the QPT's leaf nodes. During this step, the user-defined weights (i.e., ω_i from the *User SecLA*) are also populated to leaf nodes.

Following this mapping process, we will obtain one *User QPT* and one or more *CSP QPTs*[4]. The process also results in these QPTs being populated with the user-defined AND/OR and ω_i requirements.

3.3 Stage 3: Benchmarking the QPTs

Over this final stage, the actual benchmarking process relates the different "CSP QPTs" with *User QPT* as a baseline. This stage is divided in two sub-steps, namely *(1)* quantitative aggregation and, *(2)* ranking as detailed in the subsequent subsections.

3.3.1 Quantitative aggregation on the QPT

After the mapping stage (cf. Section 3.2), both the *User QPT* and the *CSP QPTs* have leaf nodes that contain values, either quantitative (e.g., cryptographic key-sizes) or qualitative (e.g., a simple "YES/NO" answer), associated with specific security attributes. Just as mentioned in [18], the use of non-homogeneous values represents a problem from the security aggregation perspective. However Casola [7] proposed a solution based on the notion of *Local Security Levels* or *LSL* (cf., Section 2.3). A LSL is defined as a function to map any class of security value (either quantitative or qualitative) to a user-defined *quantitative security level*. The concept of LSL is aligned with the notion of security ranges presented by Irvine and Levin [17] and has been widely used in the past (e.g. in [6], [8], [22] and [24]).

Our proposed methodology utilizes the notion of LSL to enable the aggregation and propagation of quantitative security values through the entire QPT. From the LSL definition [7], two basic assumptions are relevant here: *(i)* all the *i-leaf nodes* on the QPT have been already associated with a $LSL_i > 0$ and, *(ii)* there exists a maximum value LSL_{max} that is the same for *all* the leaf nodes of the QPT (i.e., $0 < LSL_i \leq LSLmax$).

Once every leaf node in the QPT has been associated with the duple $\{LSL_i, \omega_i\}$, it is possible to propagate this values to the rest of the tree using the aggregation rules shown in Table 1. These rules have been created taking into account the following:

- Within an AND node, the aggregated security level will depend on all the sibling nodes. This follows a natural notion in security, just like used in well-known scoring techniques e.g., the Common Vulnerability Scoring System (CVSS [14]).

- Following the "weakest-link" security principle, within an OR node the aggregated security level will be determined only by the sibling node with the smallest security level.

- As the notion of *LSL* and ω_i only applies to leaf nodes, the proposed aggregation rules must be slightly modified depending on the type of node being considered: *ParentL1* are located one level above leaf nodes (e.g., "CO" and "IS" in Figure 2), whereas *ParentL2* are two or more levels above leaf nodes (e.g., the root "SecLA" node in Figure 2).

From Table 1 we highlight that despite individual nodes' security levels being linearly combined (i.e., following natural security notions), the use of weights (ω_i) guarantees that all nodes will contribute to the overall security level. This is contrary to other techniques that "masquerade" the effect of some security provisions, just as discussed in Section 4.1. Future works (cf. Section 6) will research the applicability of new non-linear aggregation techniques.

Once the *User QPT* and the *CSP QPTs* have been populated with the aggregated values (quantitatively computed from Table 1), it

[4]The corresponding *QPT Template* is not needed during the rest of the proposed methodology, but can be re-used later on when processing new SecLAs based on this same template.

Table 1: Aggregation rules for a QPT with *n*-sibling nodes

	Aggregation rule with $i = 1 \ldots n$	
Parameter	AND node	OR node
$Agg_{ParentL1}$	$\sum_{i=1}^{n}(LSL_i \times \omega_i)$	$min(LSL_i \times \omega_i)$
$Agg_{ParentL2}$	$\sum_{i=1}^{n} Agg_{ParentL1,i}$	$min(Agg_{ParentL1,i})$

is possible to apply a ranking algorithm to determine how different CSPs under-/over-provision a user's requirements. This central idea behind the proposed benchmark methodology is developed next.

3.3.2 Ranking aggregated security values

In this paper, the notion of security benchmark is related to the problem of finding (either quantitatively or qualitatively) how a CSP fulfills user requirements. Our research proposes two different classes of benchmarks, namely $QuantB_{node}$ (cf. Definition 1) and $QualB_{node}$ (cf. Definition 2), both based on the quantitative security values already aggregated on the QPT (cf., Section 3.3.1).

DEFINITION 1. *The quantitative benchmark $QuantB_{node}$ associated with a specific node of the QPT, is defined as follows:*

$$QuantB_{node} = \frac{Agg_{CSP,node} - Agg_{User,node}}{Agg_{max,node}}$$

Where:

- $Agg_{CSP,node}$ *is the aggregated security value for node in the* CSP QPT*, as computed with Table 1.*
- $Agg_{User,node}$ *is the aggregated security value for node in the* User QPT*, as computed with Table 1.*
- $Agg_{max,node}$ *is is the aggregated security value for node in either* User QPT *or* CSP QPT*, as computed with Table 1 and using the maximum Local Security Level (LSL_{max}).*

The quantitative benchmark from Definition 1 is a numeric value such that $(-1 \leq QuantB_{node} \leq 1)$. If $QuantB_{node} < 0$ then the CSP is under-provisioning the user requirement, if $QuantB_{node} > 0$ then the CSP over-provisions the user requirement and finally, if $QuantB_{node} = 0$ then the CSP exactly fulfills the user requirement. In Section 4, we will show (using real-world Cloud SecLAs) that the $QuantB_{node}$ metric can be applied at any level of the QPT (including leaf nodes), thus allowing for fine-grained benchmarks.

A second metric (shown in Definition 2) aims to be more "human-friendly" by using a set of qualitative labels (e.g., *{"Copper", "Silver", "Gold"}*) to represent the benchmark's results.

DEFINITION 2. *The following expression defines $QualB_{node}$, the qualitative benchmark associated with a specific node of the QPT:*

$$QualB_{node} = \begin{cases} \lceil QuantB_{node} \times Ranks_{max} \rceil & \text{if } QuantB_{node} \geq 0 \\ \lfloor QuantB_{node} \times Ranks_{max} \rfloor & \text{if } QuantB_{node} < 0 \end{cases}$$

Where:

- $QuantB_{node}$ *is the quantitative benchmark in Definition 1.*
- $Ranks_{max}$ *is the total number of chosen qualitative labels minus one. For example, if the set of qualitative labels is {"Copper", "Silver", "Gold"} then $Ranks_{max} = 2$*

The result of the previous metric is an integer number such that $QualB_{node} = \{-Ranks_{max}, \ldots, 0, \ldots, Ranks_{max}\}$. In order to assign it a qualitative label from the set $Ranks = \{Label_1, \ldots, Label_n\}$ where $n = Ranks_{max} + 1$, we use the following mapping function:

$$f(QualB_{node} \mapsto Ranks) = \begin{cases} Label_1 & \text{if } QualB_{node} = 0 \\ Label_2 & \text{if } QualB_{node} = 1 \\ -Label_2 & \text{if } QualB_{node} = -1 \\ \vdots & \\ Label_n & \text{if } QualB_{node} = Rank_{max} \\ -Label_n & \text{if } QualB_{node} = -Rank_{max} \end{cases}$$

In the previous function, notice that a "negative" label such as $-Label_n$ literally represents the counterpart of the corresponding "positive" label $Label_n$. For example "A-" and "A+", "Silver" and "Silver-", and so forth.

Next, we present a case study that outlines how to apply the proposed benchmark methodology.

4. CASE STUDY: BENCHMARKING CLOUD PROVIDERS IN THE STAR REPOSITORY

As an initial effort to validate the proposed benchmarking methodology, we applied it to the CSP data stored in the STAR repository [10], and compared the obtained results with those empirically validated in a previous research [24] via members of the Cloud Security Alliance. Currently, STAR contains Cloud SecLAs in the form of "Consensus Assessments Initiative Questionnaire" (CAIQ [9]) reports, which provide industry-accepted ways to document what security controls exist in Cloud offerings.

The CAIQ contains a set of 171 security provisions (all of these with a qualitative "YES/NO" answer) distributed in the following *controls*: Compliance (CO) – 14, Data Governance (DG) – 15, Facility Security (FS) – 9, Human Resources Security (HR) – 4, Information Security (IS) – 71, Legal (LG) – 2, Operations Management (OP) – 5, Risk Management (RI) – 12, Release Management (RM) – 5, Resilience (RS) – 11 and Security Architecture (SA) – 23.

To perform meaningful comparisons of the obtained results with those empirically validated by Luna [24], we also use the same "extended version" of the CAIQ report containing an additional set of 29 security provisions that result in a final set of 171 + 29 = 200 security provisions to benchmark. These new security provisions allow for quantitative values (e.g., the frequency of the network penetration tests), beyond the original CAIQ's "YES/NO" answers. An overall view of the resultant QPT is shown in Figure 5.

For comparison purposes, the analyses shown in this section also considered *(i)* a maximum of 4 Local Security Levels (i.e., LSL = 4 as introduced in Section 3), *(ii)* all leaf nodes on the QPT having the same weight (i.e., ω_i), *(iii)* only *AND* branches on the QPT – except when specifying the opposite –, *(iv)* a set of qualitative labels where $Ranks_{max} = 2$ like in *Ranks = {A–, A-, A, A+, A++ }* and finally, *(v)* to fulfill STAR's usage restrictions we anonymized the identity of the benchmarked CSPs.

4.1 Analytical Results

We applied a first round of the proposed benchmarks to the same three Cloud SecLAs evaluated by Luna in [24]. The results (shown in Figure 6) have been normalized in order to allow the comparison of both approaches. Despite the differences in the obtained numeric values of the "Aggregated Security Level from SecLA" axis in Figure 6, we can observe that overall the methodology proposed in this paper also found out that CSP_2 had the worst security level. The

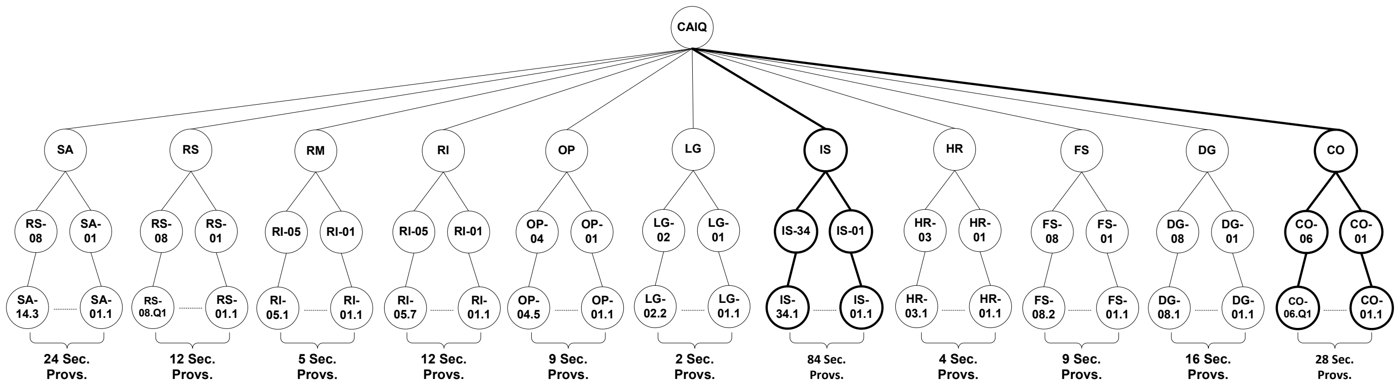


Figure 5: Quantitative Policy Tree based on the *SecLA Template* from the Cloud Security Alliance's Consensus Assessments Initiative Questionnaire [9]. Highlighted shapes are the categories used in the example shown in Section 4.

methodology proposed in this paper also eliminated REM's negative "masquerading effect" , one of the most common criticisms associated with its use. This "masquerading" is due to the flat-matrix structure used by REM (cf., Section 2.3), which does not take into account the inherent hierarchical structure of the Cloud SecLA (as elaborated in Section 2.2). When "masqueraded" the overall security score will mostly depend on those security groups with a high-number of provisions, thus affecting negatively groups with fewer — although possibly more critical — provisions. Take for example Figure 6, which clearly shows that columns "REM Technique" and "IS Policy Tree" follow the same pattern, because the CAIQ category "Information Security" (IS) is by far the one with the biggest number of security provisions in the SecLA. In Section 5 we will further discuss the main differences between the REM-based approach used in [24] and, the methodology proposed in this paper.

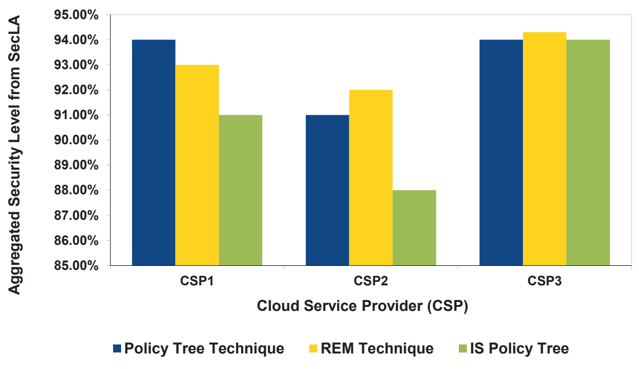


Figure 6: Quantitative comparison of the benchmarking technique proposed in this paper and the one based on REM [24].

Another improvement with respect to the REM-based approach used by Luna [24] allows users to benchmark Cloud SecLA at different levels of granularity. This is shown in Table 2, where *absolute benchmarks* were computed for three different CSPs. That is, the reference used in these benchmarks was a Cloud SecLA where all the security provisions were set to LSL = 0. Absolute benchmarks are useful to provide decision makers with a coarse-view of the security being offered by available CSPs, in order to select an initial subset where to perform the actual benchmarking with respect to their own set of security requirements.

The information shown in Table 2 is also useful to find which individual CAIQ controls report a relative CSP performance. For example, if control "CO" is the prime requisite from a business perspective, then the absolute benchmarks will advise to initially choose CSP_2 and CSP_3 over CSP_1. Notice that this conclusion can not be drawn directly from the overall SecLA-level benchmarks, where CSP_1 and CSP_3 outperform CSP_2.

Table 2: Absolute quantitative benchmarks obtained for three different *CSP SecLAs*

	CSP_1	CSP_2	CSP_3
SecLA	0.94	0.91	0.94
CO	0.67	0.92	0.89
DG	0.98	0.97	0.97
FS	1.00	0.91	1.00
HR	1.00	1.00	1.00
IS	0.91	0.88	0.94
LG	1.00	1.00	1.00
OP	1.00	0.91	0.81
RI	0.93	1.00	0.97
RM	1.00	0.70	1.00
RS	0.91	0.96	0.82
SA	0.97	0.79	0.90

A second set of benchmarks was applied to our dataset of three Cloud SecLA in order to show two additional features of the methodology proposed in this paper, namely *(i)* adding the notion of a *User SecLA* requirement – as presented in Section 3 – as a baseline for the benchmarks and, *(ii)* obtaining qualitative benchmarks that might be more meaningful than quantitative benchmarks to human decision-makers. For this analysis we created a synthetic *User SecLA* based on the QPT shown in Figure 5 with highlighted boxes. This user requirement consisted of only two controls of the CAIQ, namely "Compliance" (where all leaf nodes had $LSL_{CO} = 2$ and the same weight ω_i) and "Information Security" (where all leaf nodes had $LSL_{IS} = 3$ and also the same ω_i). The obtained results, shown in Figures 7 and 8, allow us to conclude that CSP_2 over-provisions most of the user's requirements (the "A+" bar in Figure 7) with respect to the CO control, whilst CSP_1 under-provisions almost half of them.

On the other hand, with respect to the "IS" control we found

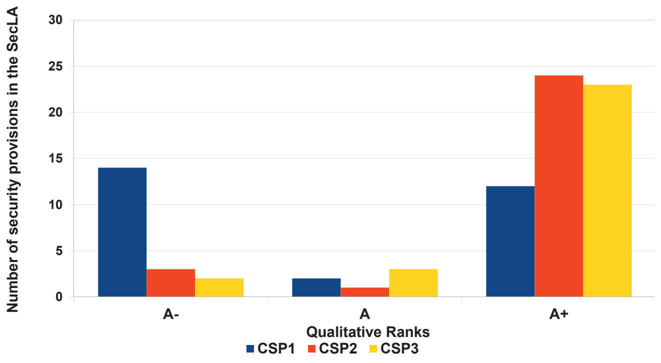


Figure 7: Qualitatively ranking a *User SecLA* using the "Compliance" category (consisting of 28 leaf nodes on the QPT) of the CAIQ [9].

that CSP_3 followed by CSP_1 over-provisioned the vast majority of user's requirements. These results are consistent with the "absolute benchmarks" from Figure 6 and Table 2, where the same couple of CSP clearly outperformed CSP_2 in the "IS" control. Also in Figure 8 is worth to notice the two negative qualitative ranks (i.e., the "A-" and "A–"), which mean that at least one provision of each CSP under-provisioned the user requirement. Users of the proposed methodology should take into account that an "A+ CSP" *does not* mean that the CSP is 100% secure or will not suffer from security failures. The proposed rankings are meant for comparing CSPs with respect to a predefined *User SecLA* requirement, therefore an "A+ CSP" means that this particular provider fulfilled the user requirements better than an "A CSP" or an "A- CSP". In Section 6, we discuss future work aimed to correlate obtained benchmarks with e.g., public reports of a CSP's outages.

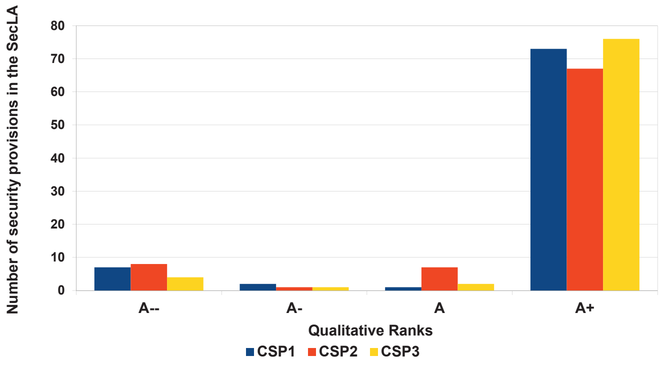


Figure 8: Qualitatively ranking a "User SecLA" using the "Information Security" category (consisting of 83 leaf nodes on the QPT) of the CAIQ [9].

Finally, Table 3 shows with a higher level of granularity both the quantitative (cf. $QuantB_{node}$ as in Definition 1) and qualitative (cf. $QualB_{node}$ as in Definition 2) benchmarks of the three CSP under analysis (due to space restrictions not all the "IS"-related provisions are being shown). Notice that we have modeled two different user requirements in the same QPT from Figure 5: the first one just considered *AND* branches and the second did consider an *OR* relationship between the top-level categories "CO" and "IS". The obtained benchmarks were once again congruent with Figures 7 and 8 and in fact, these allow for a higher level of detail in order to identify CSPs that under-provision some requirements (e.g., CSP_1 has a negative rank in category "CO-01").

Quantitative ranks, such as shown in Table 3, might also allow computer systems to automatically *match* end-users' SecLA requirements with one or more CSPs able to fulfill those needs. Despite the usefulness of such processes there is a conspicuous lack of these in research or practice. In order to bridge this gap, we next present a working prototype that implements the benchmarking methodology proposed in this paper.

4.2 Proof-of-concept: QUANTS-as-a-Service

Despite the pervasive nature of Cloud technologies and their advocated economic/technological advantages, the migration of applications has been limited, in part, due to the lack of *security assurance* by the CSP. This lack of assurance, along with the current paucity of techniques to quantify security, often results in users being unable to assess the security of the CSP they are paying for. In order to provide users with an automatic tool to benchmark the security offered by a CSP, our research contributes to the state of the art with the "QUANTifiable Security-as-a-Service" system (QUANTS-as-a-Service or simply *QUANTSaaS*).

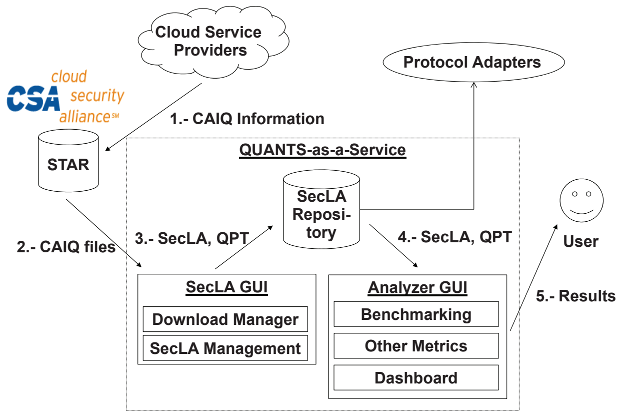


Figure 9: Building blocks of the "QUANTifiable Security-as-a-Service" system.

The first version of QUANTSaaS (shown in Figure 9) implements both the benchmarking methodology proposed in this paper and also the metrics contributed by Luna [24]. At the core of our QUANTSaaS system are the following building blocks:

1. SecLA GUI: after a CSP has uploaded its CAIQ report to STAR (Step 1 in Figure 9) the user retrieves it via the *Download Manager* (Step 2). Afterwards, both the CAIQ report and the user's required SecLA (i.e., baseline for the benchmarks including user-defined weights) are manually entered and stored into the *SecLA Repository* via the *SecLA Management* module (Step 3). This module is also used to update, delete and modify stored Cloud SecLAs. QUANTSaaS also allows for user-defined QPTs via plain-text files using the syntax shown in Figure 10.

2. SecLA Repository: this trusted database stores both the SecLAs and QPT structures used by the system. Also as future work, we will integrate a set of "protocol adapters" to directly insert/retrieve data from the SecLA Repository e.g., via CloudAudit [16].

Table 3: Example of obtained quantitative ($QuantB_{node}$) and qualitative ($QualB_{node}$) benchmarks for the categories "Compliance" (CO) and "Information Security" (IS) from CSP_1, CSP_2 and CSP_3 wrt. a *User SecLA* requirement

	CSP_1		CSP_2		CSP_3	
	$QuantB_{node}$	$QualB_{node}$	$QuantB_{node}$	$QualB_{node}$	$QuantB_{node}$	$Qual_{node}$
CO.OR.IS	0.175	A+	0.41182	A+	0.39167	A+
CO.AND.IS	0.1833	A+	0.29341	A+	0.30678	A+
CO	0.175	A+	0.425	A+	0.391	A+
CO-01	-0.25	A-	0.5	A+	0.375	A+
CO-02	0.1	A+	0.375	A+	0.35	A+
CO-03	0.075	A+	0.175	A+	0.5	A+
CO-04	0.5	A+	0.5	A+	0.5	A+
CO-05	0.5	A+	0.5	A+	0.5	A+
CO-06	0.125	A+	0.5	A+	0.125	A+
IS	0.19159	A+	0.16182	A+	0.22189	A+
IS-01	0	A	0.25	A+	0.125	A+
IS-02	0.25	A+	0.25	A+	0.25	A+
IS-03	0.25	A+	0.025	A+	0.25	A+
IS-04	0.25	A+	0.25	A+	0.25	A+
IS-05	0.25	A+	-0.5	A–	0.25	A+
IS-06	0.25	A+	0.25	A+	0.25	A+

3. Analyzer GUI: this module retrieves from the *SecLA Repository* the Cloud SecLAs to benchmark with respect to an user-defined SecLA requirement and QPT (Step 4). Depending on the metrics selected by the User (either the benchmark proposed in this paper or the metrics contributed by Luna [24]) this module will process the SecLAs. Finally, obtained results are visualized via the *Dashboard* (Step 5).

```
{AND/OR}
(
    leafNode_1
    ...
    leafNode_n
) parentNode_i
```

Figure 10: QUANTSaaS: Quantitative Policy Tree syntax.

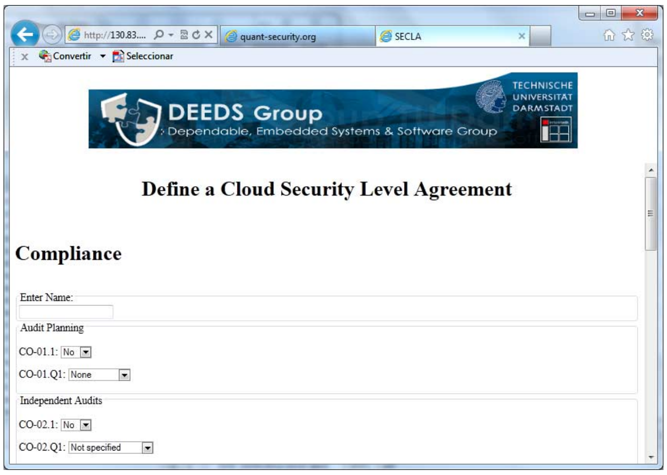
Actual screen-shots of the QUANTSaaS prototype are shown in Figures 11 and 12. The former shows the *SecLA GUI* when used to create a new CSP entry into the *SecLA Repository*. The latter presents a QTP visualization inside the QUANTSaaS' *Dashboard* showing the benchmark's results.

The QUANTSaaS system will be publicly available after its final tests have passed. Interested parties are encouraged to contact the authors of this paper for requesting access to the system, because their feedback will be used to provide further empirical validation to the methodology presented in this paper (cf., Section 6 for other future activities related with QUANTSaaS).

5. RELATED WORK

Specification and management of SLA are becoming essential components for Cloud computing. In the past, SLA specifications have mainly been considered in the Service Oriented Architectures (SOAs) and Web services fields as in WS-Agreement [2] and WSLA [21]. Unfortunately, a major limitation of these approaches is that they do not consider security aspects even if the need for incorporating security in SLAs was already highlighted by Henning [15].

While the advocacy to consider security in SLAs has started,


http://130.83....
quant-security.org
SECLA
Convertir
Seleccionar
DEEDS Group
Dependable, Embedded Systems & Software Group
TECHNISCHE UNIVERSITAT DARMSTADT
Define a Cloud Security Level Agreement

Compliance

Enter Name:

Audit Planning

CO-01.1: No

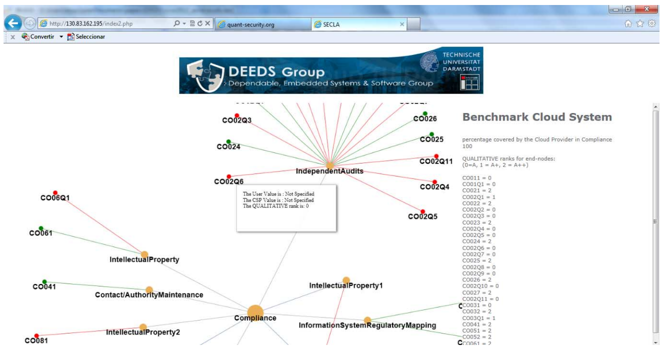
CO-01.Q1: None

Independent Audits

CO-02.1: No

CO-02.Q1: Not specified

Figure 11: QUANTSaaS: creating a new Cloud SecLA.



The User Value is : Not Specified
The CSP Value is : Not Specified
The QUALITATIVE rank is: 0

Benchmark Cloud System

percentage covered by the Cloud Provider in Compliance
100

QUALITATIVE ranks for end-nodes:
(0=A, 1 = A+, 2 = A++)

CO011 = 0
CO01Q1 = 0
CO021 = 2
CO02Q1 = 1
CO022 = 2
CO02Q2 = 0
CO02Q3 = 0
CO023 = 2
CO02Q4 = 0
CO02Q5 = 0
CO024 = 2
CO02Q6 = 0
CO02Q7 = 0
CO025 = 2
CO02Q8 = 0
CO02Q9 = 0
CO026 = 2
CO02Q10 = 0
CO027 = 2
CO02Q11 = 0
CO031 = 0
CO032 = 2
CO03Q1 = 1
CO041 = 2
CO051 = 2
CO052 = 2
CO061 = 2

Figure 12: QUANTSaaS: visualizing a Cloud SecLA benchmark via a QPT.

very few of these have focused on Cloud SecLAs per se. In particular we highlight Bernsmed [3], where the authors present a method for managing the SecLA lifecycle in the context of federated Cloud services. That work can be considered complementary to our research, considering that while Bernsmed [3] discuss the contents of Cloud SecLAs, they do not further elaborate about the techniques needed to conduct benchmarking.

Almorsy [1] proposes the notion of evaluating Cloud SecLAs. In their paper, a metric is introduced to benchmark a CSP's "security categorization" of their information (either per-tenant or per-service), based on impact metrics for all the three dimensions of security (confidentiality, integrity and availability). However, the resulting security categorization is purely qualitative (i.e. specifying the High, Medium or Low security ranges) and absolute (i.e. does not take into account the user's requirements), contrary to our proposed benchmark methodology. The benchmark methodology presented in this paper matches our previous works on the quantitative assessment of Cloud SecLAs [23], [24]. Furthermore, the methodology proposed in this paper improves over the REM-based approach (proposed in [7] and applied to Cloud SecLA in Luna [24]) by *(i)* adding the notion of *weighted* security provisions, *(ii)* providing security benchmarks at different levels of SecLA-granularity (e.g., overall Cloud SecLA or individual security provisions) and *(iii)* eliminating the "masquerading" effect mentioned in Section 4.

Despite the few works proposing new performance benchmarks for the Cloud (e.g., Binnig [4]) to the best of our knowledge there are no further works related with the security benchmarks of CSPs. In particular we refer to related works that use the notion of SecLAs and also aimed to empirically validate their security metrics with real CSP data. Nevertheless for the sake of completeness, the rest of this section cites the efforts from other Information Technology fields aimed to benchmark security.

Often the concept of security benchmark is associated with the notion of finding vulnerabilities, as discussed by Livshits and Lam [20] and implemented in their `Stanford SecuriBench` tools [19]. A similar vulnerability-based approach for security benchmarks was adopted by Parrend and Frenot [27], with a particular focus on the component-oriented OSGi platform. The authors also elaborate on a "vulnerability-pattern" to define the type of data necessary to handle OSGi vulnerabilities and also, define a quantitative metric to benchmark different OSGi implementations based on the percentage of the known vulnerabilities protected by the adopted security mechanisms.

Security benchmarks in the computer architecture area have also been studied. For example Poe and Li proposed BASS [28], a set of vulnerability-based benchmarks for evaluating the security features of hardware modifications in order to design secure architectural and hardware mechanisms.

The challenges related with the empirical validation of security benchmarks were presented by Dumitras and Shou [13]. In order to alleviate the security and privacy concerns that appear when real data is used for security benchmarks the authors propose WINE, a platform to share different types of data sets (e.g., SPAM and malware) useful to benchmark security. Taking into account the challenges documented by Dumitras and Shou [13], our research plans to make publicly available the QUANTSaaS system so future works can use our data sets and implemented techniques to validate their own security benchmark methodologies.

Finally, in a recent paper Neto [26] elaborates the difficulties involved in applying the dependability benchmarking approach to the security context. Furthermore, the authors propose as an alternative to security benchmarking the so-called "trustworthiness benchmarking", which is based on the analysis of the defense structure of the system instead of based on a set of attacks that can be conceived after a threat analysis. Our proposed methodology has followed a similar approach, by benchmarking a CSP based not on its vulnerabilities but on the quantification of their trust/assurance as documented in their Cloud SecLAs.

6. CONCLUSIONS

In this paper we have presented a methodology to benchmark Cloud security based on the concept of Security Level Agreements. The proposed methodology was applied to the CSP's information stored in the STAR repository of the Cloud Security Alliance, in order to be validated with respect to the results obtained – and empirically validated – via another state of the art Cloud security assessment technique [24]. Obtained results were congruent with the latter and furthermore, the proposed methodology also advanced the state of the art in two additional aspects: first, more fine-grained control over the performed benchmarks and second, the use of either quantitative or qualitative benchmarks.

Another contribution of our research is a working prototype that implements the proposed benchmark methodology. We refer to the QUANTaaS system described in Section 4. Our research has already considered making QUANTaaS publicly available in the short term, both to provide further empirical validation of our methodology and empower end users through providing choices of service providers via the use of Cloud SecLAs. Hopefully, our research will aid to trigger security transparency and new types of SecLA-based services in CSPs. Our belief is that SecLAs will become in the short term a key factor to enable competition between CSPs based on security properties.

To the best of our knowledge the security benchmarking of CSPs is lacking, and in particular aimed to empirically validate the applicability of their methodologies with real CSP data. Our proposed methodology specifically addresses this gap and is also flexible, in the sense that it can be used with new *Cloud SecLA Templates* that might appear in the short-term (e.g., derived from both the threat analysis of real CSP architectures and, existing standards like e.g., ISO27001 and PCI). Also, our methodology is not tied to the underlying security scoring model, because it can easily adapt to use new ones e.g., based on maturity models [29].

The work in this paper is not aimed to substitute the Cloud security auditing function. In contrast, our security benchmarks build on the premise that Cloud SecLAs are *trusted* in the sense that they have been previously audited by experts. This is precisely the base to build the *security assurance* needed by Cloud users.

We are aware of the need to provide further empirical validation of the proposed methodology, therefore future work will aim to demonstrate that our contributions can yield CSP rankings with some objective value. For example, we are planning to show that a *User SecLA* covering only the Resilience (RS) control group of the CAIQ reports, leads to a quantitative/qualitative raking of CSPs that correlates with public reports of service outages. Additional validation data is expected to come from both the QUANTSaaS users' feedback and expert-driven reviews of the CAIQ reports (e.g., possibly through our liaison with the Cloud Security Alliance). Results of the empirical validation will be also used to develop and compare with respect to new non-linear aggregation techniques and ranking algorithms, with a particular focus on those able to deal with the uncertainty and inconsistency associated with security metrics, just motivated by Jansen [18].

Also, future activities will complement both our security benchmark methodology and the QUANTSaaS system with the techniques to negotiate, predict and tune Cloud SecLAs based not only on "declarative" information (e.g., the one from the STAR repository), but also on real-time data gathered from the service provider's infrastructure. In fact, we have started the integration of our methodology into the Cloud platform developed by the EU mOSAIC project [25], in order to implement the automatic negotiation of security parameters based on the notion of Cloud SecLAs.

7. ACKNOWLEDGEMENTS

Research supported in part by EC FP7 project ABC4Trust (Grant Agreement no. 257782) and Loewe TUD CASED.

8. REFERENCES

[1] Almorsy M., *et.al.* Collaboration-Based Cloud Computing Security Management Framework. In *Proc. of IEEE Intl Conference on Cloud Computing*, pages 364–371, 2011.

[2] Andrieux K., *et.al.* Web Services Agreement Specification (WS-Agreement). Technical Report TR-WSAgreement-2007, Open Grid Forum, 2007.

[3] Bernsmed K., *et.al.* Security SLAs for Federated Cloud Services. In *Proc. of IEEE Availability, Reliability and Security*, pages 202–209, 2011.

[4] Binnig C., *et. al.* How is the weather tomorrow?: towards a benchmark for the cloud. In *Proc. of the ACM Workshop on Testing Database Systems*, pages 9:1–9:6, 2009.

[5] Bistarelli S., *et. al.* Defense trees for economic evaluation of security investments. In *Proc. of Availability, Reliability and Security*, pages 8 – 16, 2006.

[6] Casola V., et.al. Interoperable Grid PKIs Among Untrusted Domains: An Architectural Proposal. In *Advances in Grid and Pervasive Computing*, volume 4459 of *Springer LNCS*, pages 39–51. 2007.

[7] Casola V., *et.al.* A Reference Model for Security Level Evaluation: Policy and Fuzzy Techniques. *Journal of Universal Computer Science*, pages 150 – 174, 2005.

[8] Casola V., *et.al.* A SLA evaluation methodology in Service Oriented Architectures. In *Quality of Protection*, volume 23 of *Springer Advances in Information Security*, pages 119 – 130. 2006.

[9] Cloud Security Alliance. The Consensus Assessments Initiative Questionnaire. Online: https://cloudsecurityalliance.org/research/cai/, 2011.

[10] Cloud Security Alliance. The Security, Trust & Assurance Registry (STAR). Online: https://cloudsecurityalliance.org/star/, 2011.

[11] Cloud Security Alliance. Security and Privacy Level Agreements working groups. Online: https://cloudsecurityalliance.org/research/pla/, 2012.

[12] Dekker M. and Hogben G. Survey and analysis of security parameters in cloud SLAs across the European public sector. Technical Report TR-2011-12-19, European Network and Information Security Agency, 2011.

[13] Dumitras T. and Shou D. Toward a standard benchmark for computer security research: the worldwide intelligence network environment (WINE). In *Proc. of the ACM BADGERS Workshop*, pages 89–96, 2011.

[14] Forum of Incident Response and Security Teams. CVSS – Common Vulnerability Scoring System. Online: http://www.first.org/cvss/, 2012.

[15] Henning R. Security service level agreements: quantifiable security for the enterprise? In *Proc. of ACM Workshop on New security paradigms*, pages 54–60, 1999.

[16] Hoff C., *et.al.* CloudAudit 1.0 - Automated Audit, Assertion, Assessment, and Assurance API (A6). Technical Report draft-hoff-cloudaudit-00, IETF, 2011.

[17] Irvine C. and Levin T. Quality of security service. In *Proc. of ACM Workshop on New security paradigms*, pages 91–99, 2001.

[18] Jansen W. Directions in security metrics research. Technical Report TR-7564, National Institute for Standards and Technology, 2010.

[19] Livshits B. Stanford SecuriBench. Online: http://suif.stanford.edu/livshits/securibench/, 2005.

[20] Livshits B. and Lam M. Finding security errors in Java programs with static analysis. In *Proc. of Usenix Security Conference*, pages 18 – 18, 2005.

[21] Ludwig H., *et.al.* Web Service Level Agreement (WSLA) Language Specification. Technical Report TR-WSLA-2003-01-28, IBM, 2003.

[22] Luna J. *et.al.* Providing security to the Desktop Data Grid. In *Proc. of IEEE Symposium on Parallel and Distributed Processing*, pages 1–8, 2008.

[23] Luna J., *et.al.* A Security Metrics Framework for the Cloud. In *Proc. of Security and Cryptography*, pages 245–250, 2011.

[24] Luna J., *et.al.* Quantitative Assessment of Cloud Security Level Agreements: A Case Study. In *Proc. of Security and Cryptography*, (In Press).

[25] mOSAIC. mOSAIC FP7. *Online: http://www.mosaic-cloud.eu/*, 2011.

[26] Neto A., *et.al.* To benchmark or not to benchmark security: That is the question. In *Proc. of DSN HoTDep Workshop*, pages 182–187, 2011.

[27] Parrend P. and Frenot S. Security benchmarks of OSGi platforms: toward Hardened OSGi. *Softw. Pract. Exper.*, 39:471–479, 2009.

[28] Poe J. and Li T. BASS: a benchmark suite for evaluating architectural security systems. *SIGARCH Comput. Archit. News*, 34(4):26–33, 2006.

[29] Samani R., *et.al.* Common Assurance Maturity Model: Scoring Model. Online: http://common-assurance.com/, 2011.

[30] Savola R., *et.al.* Towards Wider Cloud Service Applicability by Security, Privacy and Trust Measurements. In *Proc. of IEEE Application of Information and Communication Technologies*, pages 1–6, 2010.

[31] Schneier B. Assurance. Online: http://www.schneier.com/blog/archives/2007/08/assurance.html, 2007.

[32] Weisstein W. Frobenius Norm. Online: http://mathworld.wolfram.com/FrobeniusNorm.html, 2011.

[33] Weisstein W. L1-Norm. Online: http://mathworld.wolfram.com/L1-Norm.html, 2011.