Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Novel efficient techniques for real-time cloud security assessment

Jolanda Modic [a,*], Ruben Trapero [b], Ahmed Taha [b], Jesus Luna [c,b], Miha Stopar [a], Neeraj Suri [b]

[a] XLAB d.o.o., Pot za Brdom 100, 1000 Ljubljana, Slovenia
[b] Department of Computer Science, Technische Universität Darmstadt, 64289 Darmstadt, Germany
[c] Cloud Security Alliance (Europe), Scotland, UK

ARTICLE INFO

Article history:
Received 7 December 2015
Received in revised form 5 May 2016
Accepted 20 June 2016
Available online 24 June 2016

Keywords:
Cloud security
Security quantification
Security evaluation
Security level agreements
Security metrics

ABSTRACT

Cloud computing offers multiple benefits to users by offloading them of the tasks of setting up complex infrastructure and costly services. However, these benefits come with a price, namely that the Cloud Service Customers (CSCs) need to trust the Cloud Service Providers (CSPs) with their data, and additionally being exposed to integrity and confidentiality related incidents on the CSPs. Thus, it is important for CSCs to know what security assurances the CSPs are able to guarantee by being able to quantitatively or qualitatively compare CSPs offers with respect to their own needs. On the other hand, it is also important for CSPs to assess their own offers by comparing them to the competition and with the CSCs needs, to consequently improve their offers and to gain better trust. Thus there is a basic need for techniques that address the Cloud security assessment problem. Although a few assessment methodologies have recently been proposed, their value comes only if they can be efficiently executed to support actual decisions at run time. For an assessment methodology to be practical, it should be efficient enough to allow CSCs to adjust their preferences while observing on the fly the current evaluation of CSPs' offers based on the preferences that are being chosen. Furthermore, for an assessment methodology to be useful in real-world applications, it should be efficient enough to support many requests in parallel, taking into account the growing number of CSPs and the variety of requirements that CSCs might have. In this paper, we develop a novel Cloud security assessment technique called Moving Intervals Process (MIP) that possesses all these qualities. Unlike the existing complex approaches (e.g., Quantitative Hierarchical Process – QHP) that are computationally too expensive to be deployed for the needed on-line real-time assessment, MIP offers both accuracy and high computational efficiency. Additionally, we also show how to make the existing QHP competitively efficient.

1. Introduction

The Cloud has become an important enabler of outsourcing various data storage and data processing needs for both home and business users (e.g., government, SMEs). With vastly growing number of Cloud Service Providers (CSPs), it is becoming more and more challenging for Cloud Service Customers (CSCs) to find the best provider to match not only their Quality of Service (QoS) needs, but most

* *Corresponding author.* Tel.: +38612447750.
E-mail address: jolanda.modic@xlab.si (J. Modic).
http://dx.doi.org/10.1016/j.cose.2016.06.003
0167-4048/

importantly, a provider that meets their security related requirements.

The first step in assessing and comparing CSPs is a formalization of the security properties they offer. To this end, the Cloud community is working towards a standard specification of Security Service Level Agreements (SecSLAs) (International Organization for Standardization (ISO/IEC), 2014), with which CSPs commit to the CSCs' desired level of security for the provided service. Moreover, in order to compare different SecSLAs offered by CSPs, we need a clear way to quantify the security attributes included in SecSLAs. To this end, researchers have developed a set of methodologies (e.g., Quantitative Hierarchical Process (QHP) (Taha et al., 2014), Quantitative Policy Trees (QPT) (Luna et al., 2012), Reference Evaluation Methodology (REM) (Casola et al., 2007)) to evaluate security levels guaranteed by CSPs and rank them by quantifying and comparing their SecSLAs.

However, contemporary Cloud security assessment techniques mostly focus on their use in an environment where performance is not important (e.g., in decision-making dashboards where the CSC requirements are elicited for the entire SecSLA and only subsequently evaluated). Nevertheless, it is important to develop high efficiency assessment algorithms which, by decreasing the time complexity of each assessment cycle, (i) enable processing of many requests in parallel, and (ii) facilitate CSC decisions by allowing them to select and adjust their requirements on the fly according to the current results based on what has already been chosen.

1.1. *Contributions*

To address the need for efficient Cloud security assessment algorithms, this paper makes the following contributions.

1. *Improvement of the state of the art methodology*: The first contribution is a major simplification of the well-known Cloud security assessment methodology QHP (Taha et al., 2014). We take into account properties of the methodology, apply some basic mathematical principles, and provide greatly simplified derivative termed fast QHP (fQHP) which shows a significant reduction in computational complexity.
2. *A new Cloud security assessment methodology*: The second and most important novelty of this paper is a novel high performance Cloud security assessment methodology, namely Moving Intervals Process (MIP), that focuses on ranking CSPs according to the overall quality of provided security features meeting the CSC security requirements. For each CSC request, we perform the ranking of CSPs based not only on whether they are able to fulfill with the CSC requirement or how much they under-provision, but also based on how much the request can be inherently over-provisioned to assure better global security of the acquired Cloud service.
3. *Comparison and validation*: The final part of this paper compares the original QHP methodology with its simplified version fQHP, and additionally with MIP in terms of performance. In order to validate the newly proposed Cloud security assessment approach MIP, a real-world use case is analysed.

1.2. *Paper organization*

Section 2 presents the state of the art related to Cloud security assessment. Following the description of the SecSLA model and classical security assessment processes in general in Section 3, Section 4 presents the functional analysis of the highly used QHP as a basis for developing the simplified and efficient variant fQHP. Section 5 comprises the main contribution of the paper in presenting the novel Cloud security assessment methodology MIP. The simplification of QHP is put into perspective, in Section 6, by comparison to its original version in terms of efficiency where the performance of MIP is also evaluated. Section 6 also presents the validation of MIP through a real-world use case.

2. Related work

A variety of recent research approaches target evaluating Cloud services and providers primarily for functionality with lesser emphasis on security/trust. Mostly performance and QoS indicators have captured researchers attention. One example is Rehman et al. (2011, 2012) that provided a generic model based on a Multi Criteria Decision Making (MCDM) approach to evaluate CSPs. Garg et al. (2011) compare CSPs according to QoS indicators. Menzel and Ranjan (2012) and Li et al. (2010) provide with mechanisms to evaluate performance indicators of CSPs.

The assessment techniques to evaluate CSPs have also been focused on the evaluation of trust. For example, Alabool and Mahmood (2013) advocate a trust-based model to evaluate CSPs by using fuzzy-based MCDM and linguistic descriptors. Noor and Sheng (2011) and Habib et al. (2011) evaluate trust of CSPs by taking into account the feedback received from CSCs without considering any type of qualitative or quantitative requirements.

However, in the security domain, fewer efforts exist to evaluate CSPs. Casola et al. (2006) created a methodology to evaluate the security of web service providers but it lacks of a specific vocabulary to represent the security aspects to evaluate. Frankova and Yautsiukhin (2007) addressed the aggregation of security metrics but it is focused on the analysis of business processes and the potential attacks associated to each process. A similar approach is used by Krautsevich et al. (2011) that provided with a methodology to evaluate security of services, giving more importance to the service processes rather than security features.

One of the first attempts for the evaluation of security in the Cloud computing domain was made by Almorsy et al. (2011), who proposed a model to evaluate the security of a CSP taking as an input only information about their compliance to certifications. The assessment of providers based on quantifiable security controls was initially introduced by Casola et al. (2005) with the Reference Evaluation Methodology (REM). The REM allows CSCs to express their required security levels and evaluates CSPs according to them. However, REM is limited to quantitative requirements defined for controls and is not suitable for evaluating SecSLAs with desired SLO values defined by CSCs.

A related approach was introduced by Luna et al. (2011), who proposed to quantify metrics of SecSLAs and evaluate them using a methodology called Quantitative Policy Trees (QPT) (Luna

et al., 2012). The QPT quantifies security controls based on the Cloud Control Matrix (CCM) created by the Cloud Security Alliance (CSA). The CSA CCM builds a hierarchy of security controls grouped into categories that is extended in the QPT with Service Level Objectives (SLOs). The QPT also adds AND/OR relationships between dependent controls, providing with scores based on CSC requirements that can be used to rank providers. However, the main problem with the QPT is that CSC requirements are just considered at the lowest level of the hierarchy and it is not possible to define requirements at higher levels.

The QPT was used as the foundation of another methodology, namely the Quantitative Hierarchical Process (QHP) (Taha et al., 2014) described in Section 4.1. The QHP solves the issue previously highlighted for the QPT by allowing CSCs to define qualitative requirements at any level of the SecSLA hierarchy, even if they include dependencies and conflicts (Taha et al., 2016). However, the algorithm used in the QHP is based on matrices and eigenvectors that are associated to high computational costs, hence not suitable for scenarios with efficiency constraints. A side by side comparison of the QPT and the QHP, providing insights into their individual and collective capabilities, is available in Luna et al. (2015).

Altogether, the computational complexity of these techniques has not been considered by their authors, which might be a drawback for scenarios where efficient algorithms are required (e.g., real-time assessment). The following sections of this paper address this issue by providing with two efficient techniques for evaluating security of CSPs; the first one is derived directly from the QHP and is detailed in Section 4.2, whereas the second one is a completely new technique that adds aspects not yet considered by any of the previously mentioned techniques (such as the assignment of the qualitative requirements to SLOs and taking into consideration also all higher security levels offered by CSPs with respect to CSC requirements, thus allowing the CSC to express not only one exact desired security level for each attribute but the minimal desired security level). The details of this new technique are presented in Section 5.

3. SecSLAs: terminology and assessment

In order to utilize a consistent terminology and system model, this section presents the SLA/SecSLA terminology for security assessment and also the SecSLA-based Cloud security assessment processes. The paper also utilizes the Cloud SLA structure as advocated in the ISO/IEC 19086 standard (International Organization for Standardization (ISO/IEC), 2014). According to the ETSI Cloud Standards Coordination group (European Telecommunications and Standards Institute (ETSI), 2013) SLAs should clearly specify what is being claimed for the Cloud service according to CSC requirements.

In general terms, a Cloud SLA is a contract between a CSC and a CSP that specifies the Cloud services and the security Service Level Objectives (SLOs) that the CSPs undertake to fulfill. When an SLO is not met, a violation occurs and a compensation might be required by the CSC (such as a financial compensation or an automatic service adjustment). The SLOs included in an SLA have to be quantitatively evaluated in order to perform an automatic assessment. In general, an SLO is derived from one metric (either quantitative or qualitative), where metrics are used to set the boundaries and margins of the service levels that CSPs are able to provide (along with their limitations).

A variety of schemes exist for the specification of SLAs, for example Lewis (2002), White (2000), ASP Industry Consortium (2000), and Ludwig et al. (2002). However, these typically utilize imprecise QoS indicators. Several approaches are emerging in the security domain (see International Organization for Standardization (ISO/IEC), 2013, Diver, 2007, and Swanson et al., 2005). More recently the efforts of the research community have been focused on the specification of security controls frameworks that are used to provide some degree of security assurance and transparency by providing auditors with a set of controls used to evaluate the security of a Cloud service. This is the case of the Cloud Security Alliance (CSA) with the Cloud Control Matrix (Cloud Security Alliance (CSA), 2014a). The CCM classifies security controls and organizes them into groups and categories. The CSA also provides with a questionnaire, namely the Consensus Assessments Initiative Questionnaire (CAIQ) (Cloud Security Alliance (CSA), 2014b), used by CSPs to specify their security commitments. The CSA STAR repository (Cloud Security Alliance (CSA), 2015a) compiles questionnaires from more than one hundred commercial CSPs. The interest in specifying security is growing and several academic and industrial activities are aiming at defining a common vocabulary to define security metrics (cf., ENISA (Dekker and Hogben, 2011) and NIST RATAX (National Institute of Standards and Technology (NIST), 2014)). The EC Cloud Select Industry Group on Service Level Agreements C-SIG SLA (European Commission (EC), 2014) has proposed a mapping between controls and one or more measurable security SLOs. The elicited set of SLOs and metrics can then be included into a conceptual model such as the one that the NIST Public RATAX working group (National Institute of Standards and Technology (NIST), 2014) is creating to define Security SLAs (SecSLAs).

The Cloud SecSLAs are modelled as a hierarchical structure as shown in Fig. 1. This hierarchy is the result of combining (i) the specification defined in CSA CCM to represent control groups and control categories, and (ii) the C-SIG SLA and ISO/IEC 19086 specifications to represent SLOs at the lowest level of the hierarchy.

Table 1 shows an excerpt of a SecSLA hierarchy for some examples of SLOs, taken from the extended version of the Consensus Assessments Initiative Questionnaire (CAIQ) v1.1 (Cloud Security Alliance (CSA), 2011), along with their possible values.

Security assessment methodologies use CSC requirements to evaluate the level of security assured by CSPs. Hence, the presented approach for the definition of SecSLAs can be used to map SLOs to CSPs offers and CSC requirements, thus providing a common semantic for both parties.

Naturally, the results of the assessment are reliable only if the input data is reliable. To ensure the validity of any SecSLA-based security assessment model, the SecSLAs for the considered CSPs must come from a trusted source. In practice, the trust can be assured by an external auditor performing an independent attestation of the CSPs SecSLAs (e.g., through a scheme such as the Open Certification Framework (OCF) developed by the CSA Open Certification Working group (Cloud Security Alliance (CSA), 2015b)). The audited SecSLAs are then

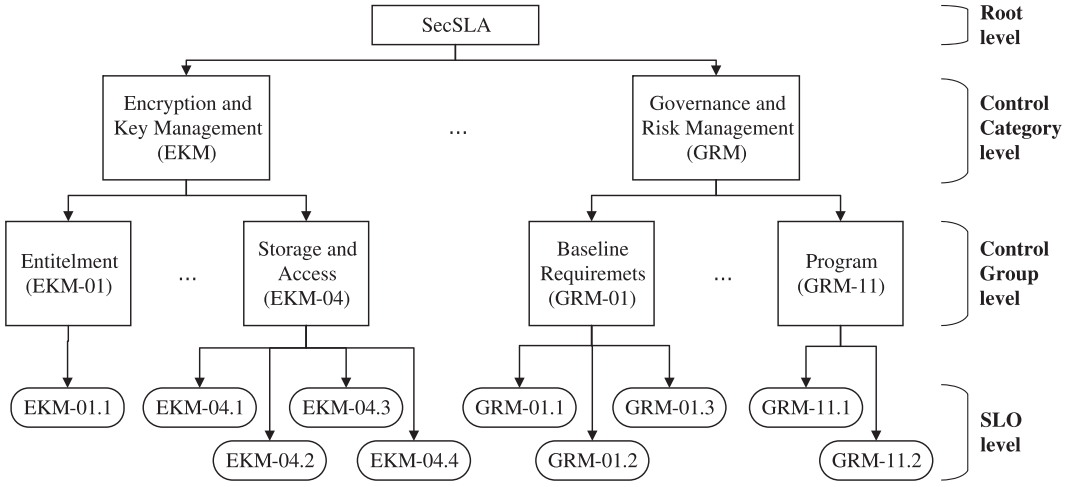

Fig. 1 – SecSLA hierarchy.

stored by the CSPs in a trusted repository of SecSLAs (e.g., the CSA STAR repository (Cloud Security Alliance (CSA), 2015a)).

The SecSLA-based assessment process comprises several progressive phases as depicted in Fig. 2. First the CSC security requirements and CSPs security provisions are gathered. In order to compare CSPs with respect to CSC's needs, requirements and offers have to be quantified. After the quantification, the evaluation is conducted at the lowest level of the SecSLA hierarchy and results are aggregated to the root level (SecSLA level) by which the final ranking of CSPs with respect to CSC requests is performed.

A typical procedure starts with elicitation (Phase 1) of CSPs security offers and CSC security requirements in the form of SecSLAs. Some techniques (e.g., the REM and the QPT) demand CSCs to express their requirements at the lowest level of the SecSLA hierarchy, whereas others (e.g., the QHP) allow CSCs to specify their security needs (either with exact values or using levels of importance) with various levels of granularity. For example, a CSC can specify an explicit value for an SLO or label it as very important, and can at the same time mark an entire control group as not important.

The second step of the process (Phase 2) encompasses quantification of CSPs provisions and CSC requests. Considering different nature of SLOs (some are qualitative with two possible values and some are quantitative with a range of possible values), each methodology introduces a different approach to quantification. The QPT and the REM use a fixed number of security levels for all SLOs, whereas QHP defines security levels for each SLO separately, depending on the type of the SLO (it uses two security levels for qualitative SLOs and maps each SLO value of a quantitative SLO to a different security level).

Table 1 – Excerpt of a concrete SecSLA hierarchy.

Control category	Control group	SLO	Description	Possible values
Compliance (*CO*)	Audit planning (*CO1*)	*CO1.1*	Production of audit assertions using a structured, industry accepted format.	*yes* > *no*
		CO1.2	Format of the produced audit assertions.	$level_3 > level_2 > level_1$
	Third party audits (*CO3*)	*CO3.1*	Permission for tenants to perform independent vulnerability scans.	*yes* > *no*
		CO3.2	External third-party vulnerability scans and penetration tests on applications and networks.	*yes* > *no*
		CO3.3	Frequency of third-party vulnerability scans and penetration tests on applications and networks.	*Monthly* > *Quarterly* > *Annual*
Risk management (*RI*)	Program (*RI1*)	*RI1.1*	Insurance by a third-party for losses.	*External* > *Internal*

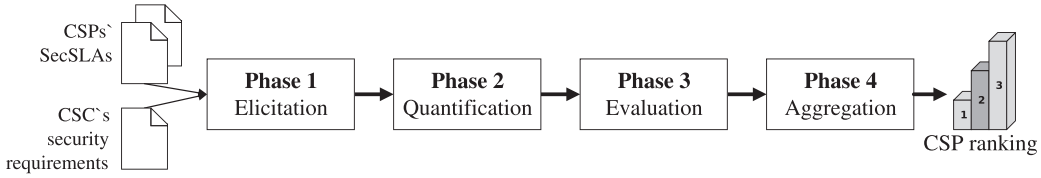

Fig. 2 – Security assessment process.

Quantified CSPs and CSC SecSLAs serve as an input for the next stage of the process (Phase 3), where each CSP SecSLA is evaluated with respect to CSC SecSLA. Different methodologies use a different evaluation algorithm. The REM defines, for each SecSLA, a matrix representing the SLO offers from a CSP as CSP SLO mapped to security levels, and evaluates the overall provision for each CSP by calculating the distance between its SecSLA matrix from the matrix corresponding to the CSC SecSLA. The QPT ranks CSPs by defining aggregated quantitative benchmarks for each SLO. The QHP introduces comparison matrices based on relative ratios among CSPs offers. The respective scores for CSPs, for each SLO, are obtained by calculating the priority vector of the corresponding comparison matrix.

The final stage (Phase 4) involves the bottom up aggregation of scores obtained on the SLO, group, and category level of the SecSLA hierarchy to rank CSPs on the group, category, and SecSLA level, respectively. Some methodologies (e.g., the QPT) rank CSPs only by calculating scores on the lowest level of the SecSLA hierarchy and aggregating them using AND/OR operations. Other techniques (e.g., the QHP) can rank CSPs on every level of the SecSLA hierarchy by aggregating scores from each level using the weighted arithmetic means approach.

4. The basic QHP and its simplification

In order to analyse and enhance the classical QHP into an efficient fQHP variant, Section 4.1 provides a basic introduction to the QHP and general insight on its limitations. These insights form the basis of simplifying the QHP operations to derive the high-efficiency variant of fQHP are presented in Section 4.2. The validation, i.e., the comparison of the QHP vs. the fQHP, appears in Section 6.1.

4.1. *Quantitative hierarchy process (QHP)*

The Quantitative Hierarchy Process (QHP), originally introduced in Taha et al. (2014), is an assessment technique that enables ranking of CSPs with respect to CSC's requirements expressed at different levels of the SecSLA hierarchy (CSPs' offers and requirements from the CSC are represented in the form of a SecSLA introduced in Section 3). The assessment is conducted in progressive stages as already discussed in Section 3.

Denoting P_i as the i-th CSP, $i = 1, 2, \ldots, n_P$, and S_k as the k-th SLO, $k = 1, 2, \ldots, n_S$, the authors of the QHP use notations presented in Table 2.

Security SLOs considered in the SecSLA can be either *boolean* (expressing whether a CSP offers a security feature or not, e.g., encryption of data at rest) or *numerical* (expressing different possible values for a security property, e.g., cryptographic key length). The QHP method handles both cases by modeling them with security levels. Assuming a numerical SLO S_k can have n different values $v_1, v_2, \ldots, v_n$, where value v_n assures the highest level of security with respect to the SLO S_k, the values are modeled as $v_i \rightarrow i$, $i = 1, 2, \ldots, n$. Let us assume that the highest security level assigned to all numerical SLOs considered in the assessment process equals N. In this case, boolean metrics are modeled as *yes*$\rightarrow N$ and *no*$\rightarrow 0$ assuming *yes* provides higher security assurance and *no*$\rightarrow N$ and *yes*$\rightarrow 0$ if *no* assures better security service.

Note that the authors (Taha et al., 2014) assume that if a CSP offers a specific value for an SLO, it is also able to provide all SLO values with lower level of granted security.

The QHP does not differentiate between a CSP that is able to grant the exact provision required by the CSC and a CSP that is able to offer even higher security levels. Thus before any calculations are done, all SLO values for all CSPs are normalized to the CSC requirements (i.e., each $V_{i,j}$ is updated to $\min(V_{i,j}, V_{U,j})$). This setting also eliminates the so called *masquerading effect* which occurs when many SLOs for which a CSP is over-provisioning, in the aggregating phase "mask" a set of SLOs for which the CSP cannot grant the CSC desired security level.

The QHP ranks CSPs by performing pairwise comparisons among all of them. The *relative rank* of CSP P_i over CSP P_j with respect to the SLO S_k is defined as

$$W_{i,j,k} = \begin{cases} 1, & \text{if } V_{i,k} = 1, \\ 0, & \text{if } V_{i,k} = 0, \end{cases} \tag{1}$$

in boolean case, and as

$$W_{i,j,k} = \begin{cases} 1, & \text{if } V_{i,k} \equiv V_{j,k}, \\ V_{i,k}/V_{j,k}, & \text{if } V_{i,k} \neq V_{j,k}, \end{cases} \tag{2}$$

in numerical case. The same formulas hold when evaluating a CSP with respect to the CSC desired value for an SLO. In the boolean case, we take:

$$W_{i,U,k} = \begin{cases} 1, & \text{if } V_{i,k} = 1, \\ 0, & \text{if } V_{i,k} = 0. \end{cases} \tag{3}$$

In the numerical case, the expression becomes:

$$W_{i,U,k} = \begin{cases} 1, & \text{if } V_{i,k} \equiv V_{U,k}, \\ V_{i,k}/V_{U,k}, & \text{if } V_{i,k} \neq V_{U,k}. \end{cases} \tag{4}$$

In order to rank CSPs for a specific SLO S_k with respect to CSC desired value for the SLO, the QHP methodology defines the *Comparison Matrix* (CM) as:

$$CM_k = \begin{bmatrix} W_{1,1,k} & W_{1,2,k} & \ldots & W_{1,n_P,k} & W_{1,U,k} \\ W_{2,1,k} & W_{2,2,k} & \ldots & W_{2,n_P,k} & W_{2,U,k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ W_{n_P,1,k} & W_{n_P,2,k} & \ldots & W_{n,n_P,k} & W_{n_P,U,k} \\ W_{U,1,k} & W_{U,2,k} & \ldots & W_{U,n_P,k} & W_{U,U,k} \end{bmatrix}. \tag{5}$$

Table 2 – QHP terms.

Term	Definition
$V_{i,k}$	Maximum value of SLO S_k provided by CSP P_i.
$V_{U,k}$	CSC required value for SLO S_k.
$W_{i,j,k}$	Relative rank of CSP P_i over CSP P_j with respect to SLO S_k.
$W_{i,U,k}$	Relative rank of CSP P_i over CSC U with respect to SLO S_k.

The relative ranking of all CSPs for the SLO S_k is defined as the normalized eigenvector (called a *Priority Vector* (PV)) of the corresponding comparison matrix CM_k.

In the final step of the QHP process, the evaluation of the overall security level offered by each CSP (and thus the final ranking of CSPs with respect to CSC requirements) is obtained by the bottom-up aggregation. This means that a PV is calculated for each element in the security SLA considering the importance weight assigned to the element by the CSC. For m elements with importance weights w_i, $i = 1, 2, \ldots, m$, to be aggregated at some hierarchy level (either at SLO, control group or control category level) we have:

$$PV_{aggregated} = [PV_1 \quad PV_2 \quad \ldots \quad PV_m] \cdot [w_1 \quad w_2 \quad \ldots \quad w_m]^T \tag{6}$$

The final PVs represent the scores for the entire SecSLAs. The last element of the final PV is the score for the CSC SecSLA and serves as the benchmark. For more details on the methodology see Luna et al. (2015) and Taha et al. (2014), or use cases considered in Section 6.

Two main issues ballast QHP, namely (i) the normalization of CSPs offers down to CSC requirements thus losing a part of information related to CSPs provisions (the methodology presented in Section 5 solves this), and (ii) the set of expensive calculations used to obtain scores for CSPs at the SLO level. When the number of CSPs increases, the size of comparison matrix for each SLO increases as well. Calculating ratios in order to form large comparison matrices and later determine their dominant eigenvectors can be computationally very expensive. Moreover, the computational cost increases drastically, when the number of SLOs increases, since the methodology has to consequently deal with a large number of large comparison matrices.

In the next subsection we show how the calculations for the QHP methodology can be simplified in order to obtain the same results but with a significantly lower time complexity.

4.2. *Fast quantitative hierarchy process (fQHP)*

Considering the basic definitions used by the authors and taking into account some basic mathematical principles, this section discusses how the original QHP methodology can be simplified. We take the definition of the comparison matrix, analyse its structure, and explicitly derive its dominant eigenvector. In result, with the fQHP we can avoid costly computations (by eliminating the need of forming comparison matrices and calculating its eigenvectors) and determine priority vectors directly from the input data (CSPs offers and CSC requirements). In the following we present a step by step simplification of the QHP.

A closer look at Eq. (1) reveals that in boolean case, since $W_{i,j,k} = 1$ if $V_{i,k} = 1$ and $W_{i,j,k} = 0$ if $V_{i,k} = 0$, the relative rank of CSP P_i over CSP P_j with respect to the SLO S_k is actually:

$$W_{i,j,k} = V_{i,k}. \tag{7}$$

Clearly, value $V_{j,k}$ has no effect on $W_{i,j,k}$, which means that the relative rank of P_i over P_j is independent of P_j. Similarly, by Eq. (3), the relative rank of CSC U over CSP P_j and the relative rank of CSP P_i over CSC U for the SLO S_k are $W_{U,j,k} = V_{U,k}$ and $W_{i,U,k} = V_{i,k}$, respectively. Since $V_{j,k}$ has no effect on $W_{U,j,k}$ and $V_{U,k}$ has no effect on $W_{i,U,k}$, value $W_{U,j,k}$ is independent to whether P_j offers SLO S_k or not and $W_{i,U,k}$ is independent to whether U requires SLO S_k or not.

Following Eqs. (5) and (7), the comparison matrix in boolean case simplifies to:

$$CM_k = \begin{bmatrix} V_{1,k} & V_{1,k} & \ldots & V_{1,k} & V_{1,k} \\ V_{2,k} & V_{2,k} & \ldots & V_{2,k} & V_{2,k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ V_{n_P,k} & V_{n_P,k} & \ldots & V_{n_P,k} & V_{n_P,k} \\ V_{U,k} & V_{U,k} & \ldots & V_{U,k} & V_{U,k} \end{bmatrix}. \tag{8}$$

The resulting CM is a matrix with identical columns. One can directly verify that the normalized priority vector for matrix (8) equals:

$$PV_k = [V_{1,k} \quad V_{2,k} \quad \ldots \quad V_{n_P,k} \quad V_{U,k}]^T / (V_{1,k} + V_{2,k} + \cdots + V_{n_P,k} + V_{U,k}). \tag{9}$$

This is due to equality

$$CM_k \cdot PV_k = (V_{1,k} + V_{2,k} + \cdots + V_{n_P,k} + V_{U,k}) \cdot PV_k$$

which means that PV_k and $V_{1,k} + V_{2,k} + \cdots + V_{n_P,k} + V_{U,k}$ form an eigenpair for the comparison matrix. Since CM_k is of rank 1 (it has only one linearly independent row and only one linearly independent column), it has only one nonzero eigenvalue. Thus, $V_{1,k} + V_{2,k} + \cdots + V_{n_P,k} + V_{U,k}$ is the only positive and thus the largest (i.e., the dominant) eigenvalue for CM_k and PV_k is the corresponding dominant eigenvector.

In numerical case, taking Eqs. (2) and (4), the comparison matrix (5) can be written as:

$$CM_k = \begin{bmatrix} 1 & V_{1,k}/V_{2,k} & \ldots & V_{1,k}/V_{n_P,k} & V_{1,k}/V_{U,k} \\ V_{2,k}/V_{1,k} & 1 & \ldots & V_{2,k}/V_{n_P,k} & V_{2,k}/V_{U,k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ V_{n_P,k}/V_{1,k} & V_{n_P,k}/V_{2,k} & \ldots & 1 & V_{n_P,k}/V_{U,k} \\ V_{U,k}/V_{1,k} & V_{U,k}/V_{2,k} & \ldots & V_{U,k}/V_{n_P,k} & 1 \end{bmatrix}. \tag{10}$$

A simple calculation shows that the i^{th} column of the matrix equals j^{th} column multiplied by $V_{i,k}/V_{j,k}$. Similarly, i^{th} row of the matrix equals j^{th} row multiplied by $V_{j,k}/V_{i,k}$. Hence, matrix (10) is of rank 1 and has exactly one nonzero eigenvalue. By taking vector (9), one can see that:

$$CM_k \cdot PV_k = (n_P + 1) \cdot PV_k.$$

Hence n_P+1 and PV_k are the dominant eigenvalue and the dominant eigenvector for the CM_k, respectively. Thus the priority vector for the matrix (10) equals the expression from (9).

In summary, in both the boolean and numerical cases, the comparison matrix has the exact same priority vector, as defined in (9). Thus, there is no need to differentiate the cases. Moreover, there is no need to calculate ratios, form matrices or calculate eigenvectors per se. In order to determine the priority vector at the SLO level, we only need the CSC required values $V_{U,k}$ for each SLO, the CSPs offered values $V_{i,k}$ for those SLOs in the form of security levels (as described at the beginning of this section), and subsequently create PVs as shown in Eq. (9). This approach is named as the fast QHP (fQHP).

In the original approach, in order to determine a PV for an SLO in the numerical case, it takes n_P^2 divisions of integers to form the comparison matrix. Additionally, determining the dominant eigenvector of a matrix is of polynomial computational complexity as well (using the power method to find approximation of the dominant eigenvector takes $\mathcal{O}(n_P^2)$ calculations per each iteration; see (Demmel, 1997)). Therefore, finding a PV for one SLO with QHP is of polynomial computational complexity.

In the simplified case of the fQHP, in order to determine a PV on the SLO level, there are no calculations required (except normalization); we only need to form a vector with known values. Note that normalization of the eigenvector is needed with both methodologies, in original and simplified, thus simplification brought us from polynomial to constant complexity (i.e., from calculating a matrix and its dominant eigenvector to only forming a vector from already known values). Considering that this reduction of time complexity applies for each SLO, the difference in performances between the QHP and the fQHP is significant.

All other steps of aggregating priority vectors at different levels of the SLA hierarchy with Eq. (6) are the same as in original approach.

With the derivation of the fQHP, we solved the time complexity weakness of the QHP methodology. However, the accuracy of the methodology can be improved, too. In the next subsection we present shortcomings of the QHP approach and present a new assessment methodology that overcomes them. Comparison of the new methodology with QHP and fQHP in terms of performance and results is discussed in Section 6.

5. Moving intervals process (MIP)

One of the characteristics of the QHP presented in Section 4.1 is that it only allows the CSC to express one desired value for each SLO. The methodology normalizes the CSPs offers down to what the CSC requires. Thus all CSPs SLO values that assure better security are ignored in the assessment process. In result, the methodology assigns the same score to two CSPs that are able to meet CSC needs, but can offer different maximum security assurances for an SLO. In case where two CSPs can meet all CSC requirements, the CSC is not given the opportunity to possibly select a CSP that is able to provide even better security guarantees as requested.

In the next subsection we provide with a new Cloud security assessment methodology, namely the Moving Intervals Process (MIP). The methodology ranks CSPs with respect to CSC requirements and levels of importance by taking into consideration (1) not only one single desired value for an SLO but the minimum required value for an SLO and (2) the exact range of security levels provided by CSPs without any normalization. With this approach the CSC is given information about which CSPs are able to meet the minimal requested security requirements. Moreover, the CSC is also informed about the extent to which the minimal requested security assurances can be over-provisioned by the CSPs. Using moving intervals, we separate scores for CSPs that are able to fulfill with CSC needs from scores of CSPs that are under-provisioning. Using this process, the CSC minimal security requirements are evaluated and serve as a benchmark which outlines CSPs that are able to not only meet the minimal requirements but are offering even higher security assurances.

By taking into account the exact CSPs provisions without normalization down to CSC requirements, the methodology assures high accuracy. Moreover, with a simple evaluation procedure, the methodology also guarantees high efficiency.

The process comprises four steps as discussed in Section 3. Compared to QHP, the elicitation phase in MIP differs in the sense that in MIP we also allow CSCs to express qualitative requirements on SLO level together with quantitative ones. In practice this means that each CSC cannot only express the exact desired minimal security level for an SLO, but can at the same time also assign an importance level to it to say how important it is that CSPs meet that requirement. Quantification phase in MIP is similar to QHP, the biggest difference between methodologies is in the evaluation and aggregation phases. In MIP all scores are based on the distance and the ratio between what the CSC requires and a CSP offers. Aggregation phase in MIP is similar to the aggregation phase in QHP in the sense that they both aggregate scores on different levels of the SecSLA hierarchy with a weighted arithmetic mean approach, but since in MIP we also consider all higher security levels provided by CSPs with respect to what the CSC requires, in MIP we perform a preprocessing step to eliminate the masquerading effect. In the following subsections we present all details for each phase.

Note that in MIP the same assumption is considered as in QHP. If a CSP assures a particular security level for an SLO, it also guarantees all lower levels.

5.1. *Phase 1: elicitation*

The process of Cloud security assessment starts with the elicitation of CSCs minimal security requirements and CSPs offers (in the form of a SecSLA introduced in Section 3). As discussed in Section 3, the SecSLA has a tree-like structure (see Fig. 1) and we use CSA STAR repository (Cloud Security Alliance (CSA), 2015a) to retrieve CSPs SecSLAs. STAR contains SecSLAs for around 140 Cloud providers in the form of *Consensus Assessments Initiative Questionnaire* (CAIQ) reports (Cloud Security Alliance (CSA), 2014b), where each of these reports currently contains 295 SLOs distributed over 222 control groups which are further organized into 16 control categories.

The set of CSC minimal security requirements is mapped to the same tree-like structure. We let the CSC to define preferred minimal SLO values (to express quantitative requirements) and/or to define levels of importance (to express qualitative security requirements) to all nodes of the SecSLA hierarchy. For example, the CSC is allowed to express minimal security requirements in terms of specific values for SLOs and is also allowed to only specify a level of importance to a certain control category or a control group without assigning specific desired minimal values for SLOs or levels of importance to nodes in the associated subtree.

Note that, as opposed to QHP, in MIP we also allow the CSC to assign importance weights on the SLO level. This way the CSC cannot only assign a specific desired value to an SLO, but can also express how important it is that a CSP matches that request. In MIP, the CSC can express requirements related to

an SLO with a specific minimal desired value or a certain level of importance or both.

5.2. *Phase 2: quantification*

As already discussed in Section 3, different SLOs can have a different number of possible values. Boolean SLOs only have two (*yes*/*no*) and numerical SLOs can have an entire range of values. We map all possible values for each SLO to security levels (represented by nonnegative integers) where the values which provide highest security assurances are mapped to higher numbers.

As in QHP, we map n different values of a numerical SLO to consecutive integers 1,2,…, n. When mapping values of boolean SLOs to security levels, we first determine the highest security level N assigned to values of all numerical SLOs considered in the assessment process. In order to make boolean and numerical SLOs comparable, we map value *no* to security level 0 and value *yes* to security level N.

For example, let us take two SLOs. For a numerical SLO *Encryption key size* with possible values $64bits < 128bits < \cdots < 1024bits < 2048bits$ the value 64*bits* is associated to security level 1 and value 2048*bits* is associated to security level 6. Since the maximum security level assigned to the only considered numerical SLO equals 6, for a boolean SLO *Encryption of Data at Rest* the values *yes* and *no* are mapped to security levels 6 and 0, respectively. In case of a numerical SLO we add level 0 which represents the case where a CSP does not offer any service associated to the SLO (e.g., for the considered example of the *Encryption key size* SLO level 0 would represent the case when a CSP does not offer any encryption).

Let us consider CSPs $P_1, P_2, \ldots, P_{n_P}$, which enforce SLOs $S_1, S_2, \ldots, S_{n_S}$, distributed over control groups $G_1, G_2, \ldots, G_{n_G}$ that are organized in security categories $C_1, C_2, \ldots, C_{n_C}$, where $n_P > 0$ and $n_S \geq n_G \geq n_C > 0$. Note that n_P, n_S, n_G, and n_C represent the number of CSPs, SLOs, control groups, and control categories considered in the assessment process. The maximum offered security level by a CSP P_i for a particular SLO S_j is denoted as $p_{i,j}$, and the CSC required minimum security level for the S_j is denoted as r_j. The maximum possible security level for an SLO S_j is denoted as N_j (hence the number of all possible levels 0,1,…, N_j for the SLO S_j equals N_j+1).

Note that the maximum possible security level N_j for the SLO S_j does not depend on CSC requirements or on CSPs offers; it solely depends on the definition of the SLO. For example, regardless of what CSPs that are included in the assessment process offer, and regardless of what the CSC requests, the *Encryption key size* SLO has 6 defined security levels as discussed above.

As for the levels of importance, we consider *high important* (*HI*), *medium important* (*MI*), and *low important* (*LI*), as in QHP. The first two are to be assigned by the CSC for the SLOs, control categories, and control group that the CSP should enforce. The level *LI* is to be assigned by the CSC for the SLOs, control categories, and control group that are out of CSC interest. Such non-relevant SLOs are excluded from the evaluation of CSPs SecSLAs.

Although CSPs and CSC SecSLA trees have the same structure, they differ in the information included. On one side, the leaf node for the SLO S_j in the SecSLA tree for the CSP P_i contains the maximum possible assured security level for the SLO $p_{i,j}$. All other nodes on higher levels of the tree are empty. On the other side, the leaf node for the SLO S_j in the CSC SecSLA tree is represented as a tuple $\{r_j, w_j^S\}$, control group nodes are represented with w_k^G, and control category nodes are represented with w_ℓ^C, where

$$w_j^{element} = \begin{cases} 1, & \text{if node is labelled as } HI, \\ 0.5, & \text{if node is labelled as } MI, \\ 0, & \text{if node is labelled as } LI, \end{cases}$$

and $element \in \{SLO, G, C\}$, as in QHP. The structure of the CSC SecSLA tree is depicted in Fig. 3.

As mentioned in Section 5.1, the CSC has an option to either express desired minimal values for all SLOs or only for some of them (with or without also assigning importance levels to them) and only assign a level of importance to some control group or control category. In order to be as precise as pos-

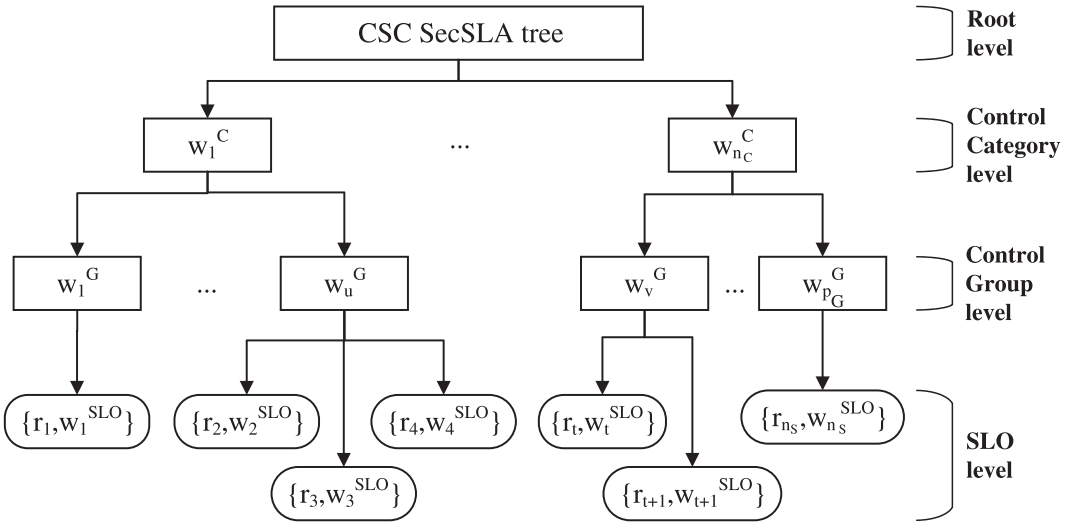

Fig. 3 – The CSC SecSLA tree.

sible and to simplify the assessment process, we want to populate all nodes in the SecSLA tree according to CSC requirements and compare all CSPs according to all SLOs, all groups, and all categories. Therefore, when a CSC only assigns a level of importance to a particular control category or control group, without specifying any desired properties for the descendant nodes, the same assigned importance weight is populated over all descendant nodes.

At the SLO level, additionally, the following mapping is performed. Let us consider a control category C_i and a descendant control group G_j that enforces SLOs $S_1, S_2, \ldots, S_k$. If the CSC assigns *HI* weight either to C_i or to G_j, all descendant SLOs are mapped to security level $\{N_\ell, 1\}$, $\ell = 1, 2, \ldots, k$, which means that in our methodology we associate *HI* weights to the highest possible security level for an SLO. In order to assure a proportional distribution of SLO values for all considered importance weights, we associate *MI* to security level $\lceil N_\ell/2 \rceil$ (we use ceiling function to assure every r_j is an integer and thus actually corresponds to some security level). Finally, importance level *LI* is assigned to security level 0. All SLOs with the importance weight 0 are excluded from the assessment process.

In summary, an SLO mapped to the control category or a control group with an importance level *HI*, *MI* or *LI* is represented by a tuple $\{N_\ell, 1\}$, $\{\lceil N_\ell/2 \rceil, 0.5\}$ or {0,0}, respectively. If the CSC specifies an exact required value for an SLO without adding the level of importance, the SLO is automatically assigned level *MI*. The same importance level *MI* is assigned to every node on the higher level of the SecSLA hierarchy, for which the CSC has not specified it.

In particular, if all numerical SLOs in the assessment process have at most 6 security levels, a boolean SLO with *HI* or *MI* importance weight is assigned values 6 or 3, respectively, and in case of *LI* the SLO is assigned value 0. A numerical SLO with, for example, 5 possible values is assigned tuples {5,1}, {3,0.5}, and {0,0} for importance levels *HI*, *MI*, and *LI*, respectively.

Such mapping of CSC requirements along the attributes of the SecSLA hierarchy guarantees that the CSC SecSLA is always full which enables an easier evaluation and aggregation. Namely in this case, we always have desired SLO values for all SLOs in the SecSLA which simplifies evaluation process. Similarly, having all importance weights specified for all nodes in the SecSLA tree simplifies aggregation process.

5.3. *Phase 3: evaluation*

The assessment process is conducted with respect to what the CSC requires. For each SLO, each provider can either meet CSC request (and possibly even over-provision it) or not. Thus for each SLO S_j we separate CSPs into two classes as follows. Class EO_j comprises all CSPs P_i that are able to provide at least the *exact* CSC minimal desired level of security for SLO S_j or are able to *over-provision* the SLO (i.e., CSPs for which $r_j \le p_{i,j}$). Class U_j comprises all CSPs that *under-provision* SLO S_j (i.e., CSPs for which $r_j > p_{i,j}$).

For example, let us consider an SLO with 6 security levels. If the CSC requires minimal security level 4, all CSPs that are able to provide at least security level 4 are considered as exact and over-provisioning CSPs (belonging to class *EO*), and CSPs that are able to assure at most level 3 are in the class of under-provisioning CSPs (i.e., in the class U_j).

By separating CSPs for each SLO S_j into two classes, we choose to have scores for CSPs on interval [0,2] and we want better fitting CSPs to have scores on the upper part of the interval. Therefore we dedicate interval [0,1) to under-provisioning CSPs and interval (1,2) to CSPs that meet CSC needs. Note that value 1 is assigned to every CSP that exactly meets CSC requirements (i.e., to every CSP for which $p_{i,j} = r_j$) and is therefore excluded from the set of possible scores for under-provisioning CSPs.

In the next two subsections we provide a detailed description of calculating scores for each class.

5.3.1. *Exact and over-provision evaluation*

Let us consider an SLO S_j with $N_j = 5$ possible security levels and let us assume that the CSC requires minimal security level $r_j = 2$. In this case the class EO_j comprises, for example, a CSP P_1 that is able to assure at most security level 2 (i.e., $p_{1,j} = 2$), a CSP P_2 that provides at most security level 3 (i.e., $p_{2,j} = 3$), a CSP P_3 offering at most security level 4 (i.e., $p_{3,j} = 4$), and a CSP P_4 that can assure all possible security levels (i.e., $p_{4,j} = 5$). These CSPs are assigned scores on interval (1, 2) where P_1 should have the lowest and P_4 the highest score. We want to equally distribute scores to all CSPs, thus considering that CSPs can over-provision 0, 1, 2, and at most 3 security levels, we assign P_1 value 1 + 0/3, P_2 value 1 + 1/3, P_3 value 1 + 2/3, and P_4 value 1 + 3/3.

In general, for each SLO we have CSPs that either provide the exact required minimal security level with no higher levels to offer (i.e., $p_{i,j} = r_j$), we have CSPs that can provide the exact required minimal security level plus one level higher (i.e., $p_{i,j} = r_j + 1$), and so on all the way up to CSPs that offer the exact requested minimal security level and also all possible higher levels for the SLO (i.e., $p_{i,j} = N_j$). This means that each CSP in class EO_j can over-provision at least 0 and at most $N_j - r_j$ security levels. To assign the score to each CSP in class EO_j, we calculate the ratio between the number of over-provisioned security levels $p_{i,j} - r_j$ and the maximum possible number of over-provisioned security levels $N_j - r_j$. Denoting $score_{EO}(P_i, S_j)$ as the score of the CSP $P_i \in EO_j$ for the SLO S_j, we obtain:

$$score_{EO}(P_i, S_j) = \begin{cases} 2, & \text{if } r_j = p_{i,j} = N_j, \\ 1 + \dfrac{p_{i,j} - r_j}{N_j - r_j}, & \text{if } r_j \le p_{i,j} \text{ and } r_j \ne N_j. \end{cases} \quad (11)$$

5.3.2. *Under-provision evaluation*

In order to demonstrate evaluation of CSPs in the under-provisioning class, we take a similar example as above. Let us take an SLO S_j with $N_j = 5$ possible security levels and in this case we assume that the CSC requires minimal security level $r_j = 3$. The class U_j includes, for example, a CSP P_1 that offers no service for the particular SLO (i.e., $p_{1,j} = 0$), a CSP P_2 that offers only security level 1 (i.e., $p_{2,j} = 1$), and a CSP P_3 that offers all security levels below the CSC request (i.e., $p_{3,j} = 2$). We want to assure proportional distribution of scores for these CSPs on the interval [0,1), and we want P_1 to have the lowest and P_3 the highest score. Therefore, considering that CSPs can under-provision 0, 1, and at most 2 security levels, where the CSC required level 3, we assign P_1 value 0/3, P_2 value 1/3 and P_3 value 2/3.

In general, for each SLO S_j we have CSPs that either do not offer any service for the SLO (i.e., $p_{i,j}=0$) or provide at most one security level less than the CSC requires (i.e., $p_{i,j}=r_j-1$). To assign scores to each CSP in the U_j class, we calculate ratio between the maximum security level $p_{i,j}$ the CSP can provide and the security level r_j that the CSC requires. Denoting $score_U(P_i,S_j)$ as the score of the CSP $P_i \in U_j$ for the SLO S_j, we have:

$$score_U(P_i,S_j)=\frac{p_{i,j}}{r_j}. \tag{12}$$

In summary, to evaluate the provision of a CSP P_i with respect to the CSC request for the SLO S_j, we merge Eqs. (11) and (12), and calculate:

$$\begin{aligned} score_S(P_i,S_j) &= \begin{cases} score_{EO}(P_i,S_j), & \text{if } P_i \in EO_j, \\ score_U(P_i,S_j), & \text{if } P_i \in U_j, \end{cases} \\ &= \begin{cases} 2, & \text{if } r_j = p_{i,j} = N_j, \\ 1+\dfrac{p_{i,j}-r_j}{N_j-r_j}, & \text{if } r_j \le p_{i,j} \text{ and } r_j \ne N_j, \\ \dfrac{p_{i,j}}{r_j}, & \text{if } p_{i,j} < r_j. \end{cases} \end{aligned} \tag{13}$$

5.4. *Phase 4: aggregation*

The aggregation phase comprises of three steps. After the assessment is concluded on the SLO level, the importance weights for SLOs are taken into account, and the scores on the SLO level are aggregated for each CSP. This way we obtain scores for the control group level. Subsequently, the aggregation is executed on the control group level to result in scores for the control category level for each CSP. Finally, in order to determine the overall SecSLA security level for each CSP, aggregation is conducted on the control category level.

The aggregation on all levels of the SecSLA hierarchy is based on the weighted arithmetic mean. In particular, for a set of values a_i with weights c_i, $i=1,2,\ldots,n$, the weighted arithmetic mean $\bar{a}_c$ is calculated as:

$$\bar{a}_c=\frac{\sum_{i=1}^{n} a_i \cdot c_i}{\sum_{i=1}^{n} c_i}. \tag{14}$$

The use of this formula may result in a masquerading effect. Let us consider a simple example. Let us take a control group G_1 that enforces SLOs S_1,S_2,S_3. Let P_1 be the provider that is assigned score $score_S(P_1,S_1)=score_S(P_1,S_2)=2$ for SLOs S_1 and S_2 labeled as *HI* by the CSC (i.e., $w_1^S=w_2^S=1$), and is assigned score $score_S(P_1,S_3)=0$ for the SLO S_3 with a *MI* label (i.e., with $w_3^S=0.5$). By Eq. (14), the weighted arithmetic mean of CSP scores on the SLO level equals:

$$\frac{score_S(P_1,S_1)\cdot w_1^S + score_S(P_1,S_2)\cdot w_2^S + score_S(P_1,S_3)\cdot w_3^S}{w_1^S+w_2^S+w_3^S}=1.60.$$

This means that even though the CSP is unable to offer the required security level for the SLO S_3, the aggregated value is still on interval (1,2) that represents CSPs that are able to grant all CSC requirements (as discussed in Section 5.3, interval (1,2) is reserved for scores for CSPs in the exact and over-provisioning class). Therefore, directly aggregating scores obtained on the SLO level, without any preprocessing, is prone to the masquerading effect on the control group level. The same observation applies to aggregating scores on all higher levels of the SecSLA hierarchy.

To eliminate the masquerading effect, we apply a correction to the scores obtained in the evaluation phase. We separate scores for the exact and over-provisioning CSPs from scores for the under-provisioning CSPs. Instead of using the initial interval (1,2) for scores for the exact and over-provisioning CSPs (on SLO level), we increase the distance between (1,2) and interval [0,1) reserved for scores for under-provisioning CSPs (on SLO level), and move it to some interval $[y,y+1]$, $y>1$, as discussed below.

A similar move of the interval with scores for the exact and over-provisioning CSPs is conducted on control group and control category level. All details about how we move intervals are provided in the following subsections dedicated to each step of the aggregation process.

5.4.1. *SLO level*

In order to aggregate scores at the SLO level and at the same time eliminate the masquerading effect, the first step is to separate intervals. Considering that we can have at most n_S values to aggregate for each CSP on the SLO level, we increase the distance between intervals [0,1) and (1,2) for n_S, where n_S is the number of all SLOs considered in the SecSLA hierarchy.

Instead of using the initial interval (1,2), we move it to $[1+n_S, 2+n_S]$. In particular, before the aggregation process, we update each score $score_S(P_k,S_j)$ that is on interval (1,2) to $score_S(P_k,S_j)+n_S$. In more formal mathematical notation this means that we map each score $score_S(P_k,S_j)$ that lies on interval (1,2) to $score_S(P_k,S_j)+n_S$, i.e.:

$$score_S(P_k,S_j)\in[1,2] \mapsto score_S(P_k,S_j)+n_S. \tag{15}$$

After this update, all under-provisioning CSPs have scores on interval [0,1) and all exact and over-provisioning CSPs have scores on interval $[1+n_S, 2+n_S]$.

The interested reader can use Eq. (14) and see that (1) regardless of the number of SLOs a CSP is able to over- or under-provision, and (2) regardless of the weights assigned to them, after aggregating the values, the scores on interval $[1+n_S, 2+n_S]$ can only be associated with CSPs that are able to exact or over-provision all considered SLOs. The remaining CSPs that can either offer the requested security level for some or none SLOs have lower scores.

The next step of the aggregation process on the SLO level is to calculate the weighted arithmetic mean of the updated scores. Let us consider a control group G_i which enforces SLOs $S_1,S_2,\ldots,S_j$. For the CSP P_k the aggregation of the scores on the SLO level is performed by the equation:

$$score_G(P_k,G_i)=\frac{\sum_{t=1}^{j} score_{SLO}(P_k,S_t)\cdot w_t^S}{\sum_{t=1}^{j} w_t^S}. \tag{16}$$

The resulting $score_G(P_k,G_i)$ represents the score for the CSP P_k for the control group G_i.

Note that since we aggregate values from intervals [0,1) and $[1+n_S, 2+n_S]$, all obtained scores on the control group level are on interval $[0,2+n_S]$. In particular, all scores for the CSPs that cannot fulfill all CSC requirements on the control group level are on interval $[0,1+n_S)$, and scores for the exact and over-provisioning CSPs are on interval $[1+n_S, 2+n_S]$.

5.4.2. *Control group level*

The masquerading effect can also appear when aggregating values on the control group level. In order to eliminate it, we need to separate interval representing values for exact and over-provisioning CSPs from the rest. Considering that we can have at most n_G values to aggregate for each CSP on the control group level (we have n_G control groups altogether), we increase the distance between intervals $[0,1+n_S)$ and $[1+n_S, 2+n_S]$ for n_G, where n_G is the number of all control groups in the SecSLA hierarchy.

Instead of aggregating scores on intervals $[0,1+n_S)$ and $[1+n_S, 2+n_S]$, we move the latter one to the interval $[(1+n_S)+n_G, (2+n_s)+n_G]$. In practice, before the aggregation process on the control group level, we update each score $score_S(P_k, G_i)$ that is on interval $[1+n_S, 2+n_S]$ to score $score_S(P_k, G_i)+n_G$. In more formal mathematical notation this means that we map each score $score_S(P_k, G_i)$ that lies on interval $[1+n_S, 2+n_S]$ to score $score_S(P_k, G_i)+n_G$, i.e.:

$$score_S(P_k, G_i) \in [1+n_S, 2+n_S] \mapsto score_S(P_k, G_i)+n_G. \tag{17}$$

After this mapping, all CSPs that are able to fulfill CSC requirements for all control groups have scores on interval $[1+n_S+n_G, 2+n_S+n_G]$ and the remaining CSPs have scores on interval $[0,1+n_S)$.

An interested reader can use Eq. (14) and confirm that regardless of the number of control groups a CSP is able to over- or under-provision, and regardless of the weights assigned to them, after aggregating values, scores on interval $[1+n_S+n_G, 2+n_S+n_G]$ can only be associated with CSPs that are able to fully provision all considered control groups. The remaining CSPs have lower scores.

Let us now consider a control category C_ℓ that enforces control groups $G_1, G_2, \ldots, G_j$. Aggregation of the updated values on the control group level for the CSP P_k is conducted by the equation:

$$score_C(P_k, C_\ell) = \frac{\sum_{t=1}^{j} score_G(P_k, G_t)\cdot w_t^G}{\sum_{t=1}^{j} w_t^G}. \tag{18}$$

The resulting $score_C(P_k, C_\ell)$ represents the score for the CSP P_k for the control category C_ℓ.

Note that by averaging values from intervals $[0,1+n_S)$ and $[1+n_S+n_G, 2+n_S+n_G]$, all obtained scores on the control category level are on interval $[0, 2+n_S+n_G]$. In particular, CSPs that cannot fulfill all CSC requirements have scores on interval $[0, 1+n_S+n_G)$, and all exact and over-provisioning CSPs have scores on interval $[1+n_S+n_G, 2+n_S+n_G]$.

5.4.3. *Control category level*

The final step of the process is aggregation of scores on the control category level. Similarly as for scores on SLO and control group level, we preprocess scores on the control category level to avoid the masquerading effect. Considering that we can have at most n_C values to aggregate for each CSP on the control category level, we increase the distance between intervals $[0, 1+n_S+n_G)$ and $[1+n_S+n_G, 2+n_S+n_G]$ for n_C, where n_C is the number of all control categories in the SecSLA hierarchy.

Instead of using intervals $[0, 1+n_S+n_G)$ and $[1+n_S+n_G, 2+n_S+n_G]$, we move the latter one to the interval $[(1+n_S+n_G)+n_C, (2+n_S+n_G)+n_C]$. This means that before the aggregation process on the control category level, we update each score $score_S(P_k, C_\ell)$ that is on interval $[1+n_S+n_G, 2+n_S+n_G]$ to score $score_S(P_k, C_\ell)+n_C$. In a formal mathematical notation this means that we map each score $score_S(P_k, C_\ell)$ that lies on interval $[1+n_S+n_G, 2+n_S+n_G]$ to score $score_S(P_k, C_\ell)+n_C$, i.e.:

$$score_S(P_k, C_\ell) \in [1+n_S+n_G, 2+n_S+n_G] \mapsto score_S(P_k, C_\ell)+n_C. \tag{19}$$

After this mapping, all CSPs that are able to fulfill CSC requirements for all control categories have scores on interval $[1+n_S+n_G+n_C, 2+n_S+n_G+n_C]$ and the remaining CSPs have scores on interval $[0, 1+n_S+n_G)$.

A more interested reader can use Eq. (14) and verify that regardless of the number of control categories a CSP is able to over- or under-provision, and regardless of the weights assigned to them, after aggregating values, scores on interval $[1+n_S+n_G+n_C, 2+n_S+n_G+n_C]$ can only be assigned to CSPs that are able to fully provision all considered control categories. All remaining CSPs have lower scores.

Let us assume that the SecSLA enforces control categories $C_1, C_2, \ldots, C_j$. The overall security level of the SecSLA for the CSP P_k is determined by the equation:

$$score(P_k) = \frac{\sum_{t=1}^{j} score_C(P_k, C_t)\cdot w_t^C}{\sum_{t=1}^{j} w_t^C}. \tag{20}$$

The resulting $score(P_k)$ represents the score for the CSP P_k for the entire SecSLA.

Note that by calculating the weighted arithmetic mean of values from intervals $[0, 1+n_S+n_G)$ and $[1+n_S+n_G+n_C, 2+n_S+n_G+n_C]$, the obtained scores on the SecSLA level are on interval $[0, 2+n_S+n_G+n_C]$. In particular, CSPs that cannot fulfill all CSC requirements have scores on interval $[0, 1+n_S+n_G+n_C)$, and all exact and over-provisioning CSPs have scores on interval $[1+n_S+n_G+n_C, 2+n_S+n_G+n_C]$.

5.4.4. *Normalization*

At the end of the aggregation process we normalize all scores by dividing them with $2+n_S+n_G+n_C$. For the purpose of understanding the obtained results, we take as a benchmark the normalized lower boundary $(1+n_S+n_G+n_C)/(2+n_S+n_G+n_C)$ of scores for CSPs that fulfill all CSC security requirements. All CSPs that have final scores above this baseline are the ones that are able to grant all security requirements expressed by the CSC.

The proposed security assessment methodology is demonstrated through a real-world use case in Section 6.2.

5.5. *Performance (time complexity) analysis*

We conclude this section with a brief comparison of time complexities for fQHP and MIP.

By Eq. (13), the MIP methodology requires for each SLO S_j 1 operation to calculate the divisor $d = N_j - c_j$ and additional $3n_P$ operations to calculate scores $2+(p_{i,j}-r_j)/d$ for all CSPs P_i. In the worst case, this means $n_S \cdot (1+3n_P) = 3n_P n_S + n_S$ operations for all SLOs and all CSPs.

With fQHP the first step is normalization of all CSPs offers down to CSC requirements. For n_P CSPs and n_S SLOs this takes at most $n_P \cdot n_S$ operations. To normalize each PV for each SLO S_k in Eq. (9), it requires n_P operations to calculate the divisor $d = V_{1,k} + V_{2,k} + \cdots + V_{n_P,k} + V_{U,k}$ and further n_P+1 operations to perform all divisions $V_{1,k}/d, V_{2,k}/d, \ldots, V_{n_P,k}/d, V_{U,k}/d$. For all SLOs this results in $n_S \cdot (2n_P + 1)$ for normalization of PVs for all SLOs. Thus fQHP requires altogether $n_P \cdot n_S + n_S \cdot (2n_P + 1) = 3n_P n_S + n_S$ operations.

This means that in the worst case, MIP and fQHP have the same time complexity. Any differences in performance are due to the CSC requirements and the number of CSPs that are able to fulfill them. In particular, if many CSPs are under-provisioning a large number of SLOs, MIP will be more efficient than QHP since calculating scores on SLO level with MIP will by Eq. (13) only require 1 single operation (to calculate $p_{i,j}/r_j$) instead of 3 (to calculate $1+(p_{i,j}-r_j)/(N_j-r_j)$) for each SLO and each under-provisioning CSP.

Comparison of fQHP and MIP in terms of performance in practice is presented in Section 6.1.

6. Validation

This section presents the validation of both newly proposed Cloud security assessment approaches discussed in this paper, namely the fQHP and the MIP.

First we compare the QHP with its improved version fQHP in terms of performance. Additionally we evaluate efficiency of the MIP with respect to the fQHP. Afterwards we focus on the validation of the MIP approach through a set of different use cases. We compare results obtained by the MIP with respect to the QHP and discuss accuracy.

Experiments presented in this paper are conducted with Matlab R2015b on Intel i5 CPU, 2.5 GHz, and 8 GB RAM.

6.1. *Efficiency of fQHP and MIP*

In order to evaluate the performance of the methodology fQHP introduced in Section 4.2 with respect to the original algorithm, and to compare it to the newly proposed methodology MIP discussed in Section 5, we take $n_P = 150$ CSPs that offer $n_S = 300$ SLOs with different numbers of possible values (each SLO has a random number of possible values ranging from 2 to 6). We take one CSC which expresses a random desired security level for each SLO.

Since the QHP and the fQHP differ only in the evaluation of the CSPs security levels at the SLO level of the SecSLA hierarchy, comparison of performances is conducted at the SLO level only. This means that with each methodology only the scores at the SLO level are calculated (where all SLOs are equally important) without any aggregations at SLO or higher levels.

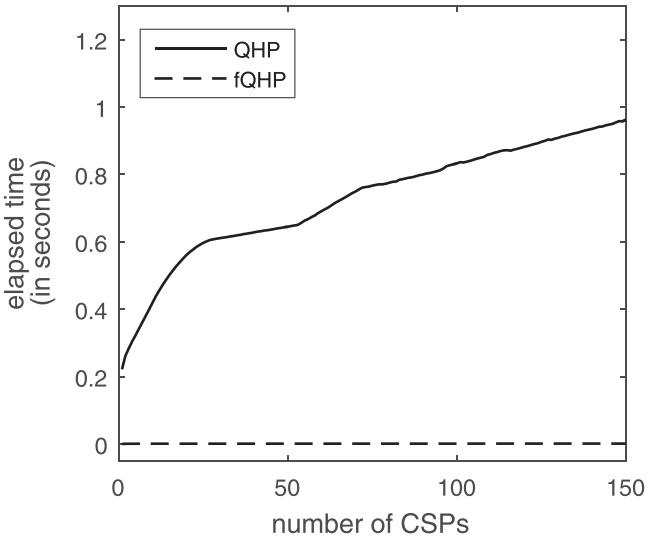

Fig. 4 – Comparison of QHP and fQHP with respect to increasing number of CSPs.

As discussed in Section 4.2, the complexity of the QHP methodology increases with a growing number of CSPs (size of the comparison matrix, for which the dominant eigenvector has to be determined, depends on the number of CSPs). Therefore we compare the QHP and the fQHP with respect to a fixed number of SLOs inside one control group and different number of CSPs. For each number k of CSPs, $k = 1, 2, \ldots, 150$, we measure the time required to evaluate corresponding SLAs for all 300 SLOs. We use inbuilt functions for calculation of the dominant eigenvector of each comparison matrix. Results are presented in Fig. 4.

If we fix the number of CSPs (i.e., we take a large number of CSPs), the performance of the QHP is also affected by an increasing number of SLOs in the sense that it requires more time to calculate an additional priority vector for a large comparison matrix with each added SLO. In Fig. 5 we present comparison of times required to evaluate all CSP SecSLAs on the SLO level with the QHP and the fQHP for an increasing number of SLOs. In particular, for 150 CSPs we measure the time required to evaluate corresponding SecSLAs for k SLOs, where $k = 1, 2, \ldots, 300$.

For better presentation of results, we have smoothed data in all graphs using moving average filter of span 20.

Simplifications clearly improve the performance of the QHP methodology. For the cases considered above, performance of the QHP is, in average, improved by a factor 500. The performance is improved to the point where the methodology can be used for the real-time assessment during the process of elicitation of the CSC security requirements.

In order to evaluate the performance of our proposed Cloud security assessment algorithm MIP (Section 5), we now compare the fQHP and the MIP with respect to the increasing number of CSPs and SLOs in Fig. 6 and Fig. 7, respectively. Since the fQHP and the MIP differ also in the quantification phase, in each

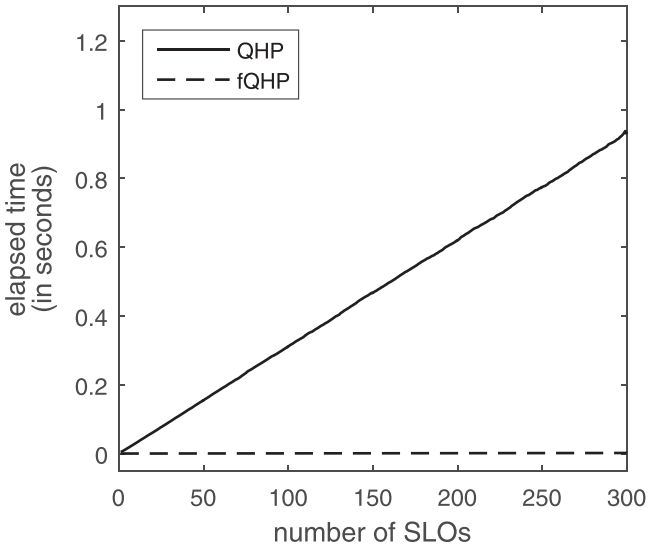

Fig. 5 – Comparison of QHP and fQHP with respect to increasing number of SLOs.

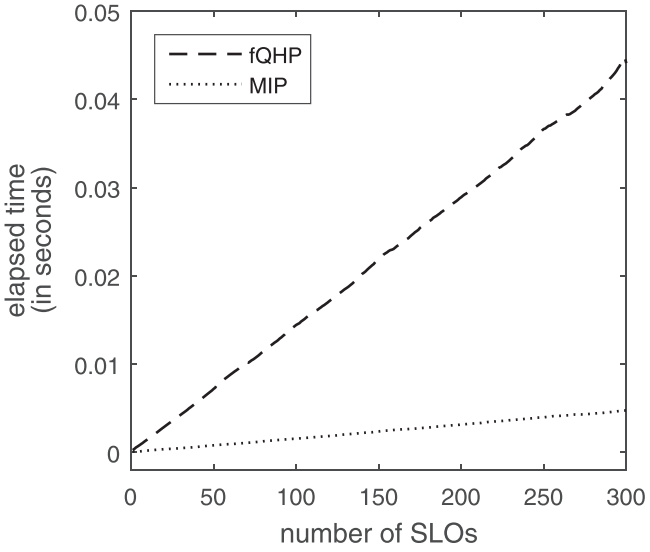

Fig. 7 – Comparison of fQHP and MIP with respect to increasing number of SLOs.

considered case we compare the time required to (a) normalize CSPs offers down to CSC requirements and to (b) determine priority vectors with fQHP, to the time required to calculate scores with MIP.

The MIP methodology not only considers exact CSPs offers (as oppose to the QHP and the fQHP which only take into account CSPs provisions normalized to the CSC requirements), but as seen in Figs 6 and 7, it also assures better overall performance. In the set of conducted tests, the MIP is, in average, around 9 times faster than the fQHP.

In summary, performance figures show that one single request to evaluate 150 SecSLAs (CSAs STAR repository currently contains around 150 SecSLAs) with 300 SLOs (CSAs CAIQ currently contains around 300 SLOs) is processed in ~0.045s when using the fQHP algorithm and in ~0.005s when using the MIP methodology. These numbers imply that we can process more than 20 requests per second with fQHP and around 200 requests per second with MIP, which demonstrates usability of the approaches in real-time assessment.

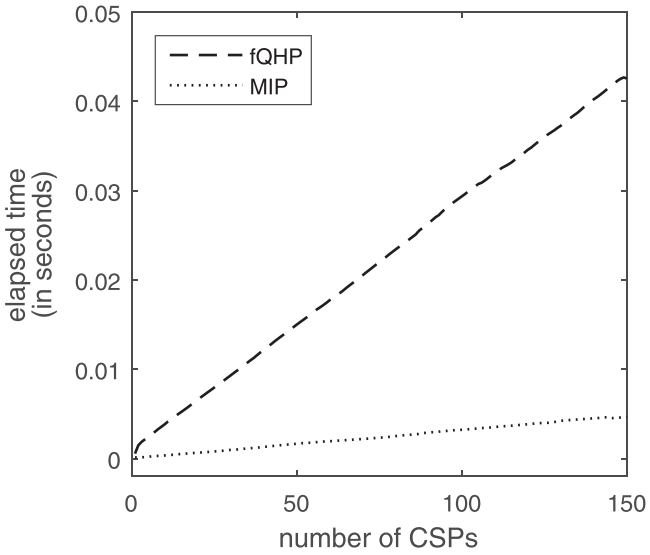

Fig. 6 – Comparison of fQHP and MIP with respect to increasing number of CSPs.

6.2. *Accuracy of MIP*

In order to validate the MIP methodology, we consider an example introduced in Luna et al. (2015) and analyse the results obtained by the MIP with respect to results obtained by the QHP. Table 3 presents the sample dataset with both boolean and numerical SLOs and three different CSCs; the CSP P_1 that specifies requirements on SLO level, P_2 that specifies requirements on all levels of the SecSLA hierarchy, and P_3 that only assigns the importance weights on the category level.

In the considered example, we have 12 boolean and 4 numerical SLOs taken from the extended version of the Consensus Assessments Initiative Questionnaire v1.1 (Cloud Security Alliance (CSA), 2011) (some are described in Table 1). Boolean SLOs have values *yes* and *no*. SLOs *CO*1.2 and *FS*2.2 are related to different formats of audit assertions and to the process of background verification of employees, contractors, and third parties, respectively, thus all possible values for these SLOs are expressed in terms of levels (e.g., $level_2$, $level_3$). SLO *CO*3.3 is associated to the frequency of third party audits, hence having possible values *Annual*, *Quarterly*, and *Monthly*, and SLO *RI*1.1 is related to insurance for losses due to outages, thus having possible values *Internal* and *External*.

Note that values *Annual*, *Quarterly*, and *Monthly* for the SLO *CO*3.3 are mapped to security levels 2, 3, and 4, respectively. Values *Internal* and *External* for the SLO *RI*1.1 are mapped to levels 2 and 1, respectively. All other SLOs are either numerical with values $level_i$ mapped to levels i or boolean, where value *yes* is mapped to the highest security level i considered in the assessment process (in our case, to security level 4) and *no* is mapped to level 0.

Table 3 – Excerpt of CSPs SecSLAs and CSCs security requirements as introduced in Luna et al. (2015).

Control category ID	Control group ID	SLO ID	P_1	P_2	P_3	CSC_1	CSC_2	CSC_3
CO	CO1	CO1.1	Yes	Yes	Yes	Yes	HI	HI
		CO1.2	$Level_3$	$Level_2$	$Level_3$	$Level_3$		
	CO2	CO2.1	No	Yes	Yes	Yes	LI	
		CO2.2	Yes	Yes	Yes	Yes		
		CO2.3	Yes	Yes	Yes	Yes		
		CO2.4	Yes	Yes	Yes	Yes		
	CO3	CO3.1	No	Yes	Yes	Yes	MI	
		CO3.2	Yes	Yes	Yes	Yes		
		CO3.3	Quarterly	Annual	Monthly	Monthly		
FS	FS1	FS1.1	No	Yes	Yes	Yes	HI	LI
		FS1.2	Yes	No	Yes	Yes		
	FS2	FS2.1	Yes	Yes	Yes	Yes		
		FS2.2	$Level_3$	$Level_2$	$Level_3$	$Level_3$		
		FS2.3	Yes	Yes	Yes	Yes		
RI	RI1	RI1.1	Internal	Internal	External	Internal	Internal	MI
		RI1.2	Yes	Yes	Yes	Yes	No	

First step of the assessment process is quantification of CSPs security offers and CSCs security requirements as discussed in Section 5.2. Results of the quantification process for all CSPs and CSCs are presented in Table 4.

Since the authors in (Luna et al., 2015) do not consider the importance weights at the SLO level, for CSC_1 we consider them as all equally important and label them as *MI*. For any element in the SecSLA hierarchy that is not assigned an importance weight by the CSC and is not assigned the importance weight by inheritance, we label it as *MI*.

In the considered example, the CSC_1 specifies desired SLO values on SLO level. To those we add the *MI* importance levels. We additionally assign *MI* levels to all control groups that enforce chosen SLOs and to all control categories that enforce chosen groups.

The CSC_2 specifies exact required values for a couple of SLOs (namely *RI*1.1 and *RI*1.2), assigns levels of importance to some control groups (namely *CO*1, *CO*2, and *CO*3), and labels one entire control category (namely *FS*) as high important. In this case, all groups and all SLOs in the *FS* category deduce the *HI* importance level. All SLOs in the *CO* control groups deduce importance levels assigned to those groups. All other nodes in the SecSLA are labeled as *MI*. The security levels for all unspecified SLOs are defined as proposed in Section 5.2.

The last CSC, namely CSC_3, expresses security requirements only at the control category level by assigning them specific importance levels. In this case, all control groups and all SLOs enforced by the chosen control categories deduce the specified importance levels, and security levels for SLOs are defined as discussed in Section 5.2.

Let us now focus on CSC_1. We denote SV_k^S as the score vector for CSP P_k where its j-th element represents score for P_k with respect to SLO S_j (i.e., contains value $score_s(P_k, S_j)$), $j = 1, 2, \ldots, 16$. By Eq. (13) (and columns N_j, $p_{1,j}$, $p_{2,j}$, $p_{3,j}$, and r_j for CSC_1 in Table 4), we obtain the following score vectors for all three CSPs:

Table 4 – Quantified CSPs security provisions and CSCs security requirements.

				P_1	P_2	P_3	CSC_1				CSC_2				CSC_3			
C_ℓ	G_i	S_j	N_j	$p_{1,j}$	$p_{2,j}$	$p_{3,j}$	r_j	w_j^S	w_i^G	w_ℓ^C	r_j	w_j^S	w_i^G	w_ℓ^C	r_j	w_j^S	w_i^G	w_ℓ^C
CO	CO1	CO1.1	4	4	4	4	4	0.5	0.5	0.5	4	1	1	0.5	4	1	1	1
		CO1.2	3	3	2	3	3	0.5			3	1			3	1		
	CO2	CO2.1	4	0	4	4	4	0.5	0.5		0	0	0		4	1	1	
		CO2.2	4	4	4	4	4	0.5			0	0			4	1		
		CO2.3	4	4	4	4	4	0.5			0	0			4	1		
		CO2.4	4	4	4	4	4	0.5			0	0			4	1		
	CO3	CO3.1	4	0	4	4	4	0.5	0.5		2	0.5	0.5		4	1	1	
		CO3.2	4	4	4	4	4	0.5			2	0.5			4	1		
		CO3.3	4	3	2	4	4	0.5			2	0.5			4	1		
FS	FS1	FS1.1	4	0	4	4	4	0.5	0.5	0.5	4	1	1	1	0	0	0	0
		FS1.2	4	4	0	4	4	0.5			4	1			0	0		
	FS2	FS2.1	4	4	4	4	4	0.5	0.5		4	1	1		0	0	0	
		FS2.2	3	3	2	3	3	0.5			3	1			0	0		
		FS2.3	4	4	4	4	4	0.5			4	1			0	0		
RI	RI1	RI1.1	2	1	1	2	1	0.5	0.5	0.5	1	0.5	0.5	0.5	1	0.5	0.5	0.5
		RI1.2	4	4	4	4	4	0.5			0	0.5			2	0.5		

$SV_1^S = [2.00, 2.00, 0.00, 2.00, 2.00, 2.00, 0.00, 2.00, 0.75, 0.00, 2.00, 2.00, 2.00, 2.00, 1.00, 2.00]$

$SV_2^S = [2.00, 0.67, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 0.50, 2.00, 0.00, 2.00, 0.67, 2.00, 1.00, 2.00]$

$SV_3^S = [2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00, 2.00]$

As discussed in Section 5.4.1, in order to avoid the masquerading effect, we update all values $score_S(P_k, S_j)$ on interval (1,2) to values $score_S(P_k, S_j) + n_S = score_S(P_k, S_j) + 16$ by rule (15). Updated scores are:

$SV_1^S = [18.00, 18.00, 0.00, 18.00, 18.00, 18.00, 0.00, 18.00, 0.75, 0.00, 18.00, 18.00, 18.00, 18.00, 17.00, 34.00]$

$SV_2^S = [18.00, 0.67, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 0.50, 18.00, 0.00, 18.00, 0.67, 18.00, 17.00, 34.00]$

$SV_3^S = [18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00, 18.00]$

We now use the obtained score vectors, column w_j^S for CSC_1 in Table 4, and Eq. (16) to obtain scores on the control group level. We obtain the following score vectors SV_k^G for all CSPs P_k for the group level, where i-th element represents sore for P_k with respect to control group G_i (i.e., contains value $score_G(P_k, G_i)$), $i = 1, 2, \ldots, 6$.

$SV_1^G = [18.00, 13.50, 6.25, 9.00, 18.00, 17.50]$

$SV_2^G = [9.33, 18.00, 12.17, 9.00, 12.22, 17.50]$

$SV_3^G = [18.00, 18.00, 18.00, 18.00, 18.00, 18.00]$

Before performing the next aggregation, we update the values $score_G(P_k, G_i)$ that lie on the interval (17,18) to $score_G(P_k, G_i) + n_G = score_G(P_k, G_i) + 6$ by Eq. (17). Updated scores are:

$SV_1^G = [24.00, 13.50, 6.25, 9.00, 24.00, 23.50]$

$SV_2^G = [9.33, 24.00, 12.17, 9.00, 12.22, 23.50]$

$SV_3^G = [24.00, 24.00, 24.00, 24.00, 24.00, 24.00]$

To obtain scores on the control category level, we use the score vectors SV_k^G, column w_i^G for CSC_1 in Table 4, and Eq. (18). The resulting score vectors SV_k^C for all CSPs P_k for the category level, where ℓ-th element represents the score for the P_k with respect to the category C_ℓ, are:

$SV_1^C = [14.58, 16.50, 23.50]$

$SV_2^C = [15.17, 10.61, 23.50]$

$SV_3^C = [24.00, 24.00, 24.00]$

Before the final aggregation, we use Eq. (19) and update values $score_C(P_k, C_j)$ that lie on the interval [23,24] to $score_C(P_k, C_j) + n_C = score_C(P_k, C_j) + 3$. The updated scores are:

$SV_1^C = [14.58, 16.50, 26.50]$

$SV_2^C = [15.17, 10.61, 26.50]$

$SV_3^C = [27.00, 27.00, 27.00]$

By taking score vectors SV_k^C, column w_i^C for CSC_1 in Table 4, and Eq. (20), we obtain the final evaluations of CSPs SecSLAs. The resulting scores are $score(P_1) = 19.19$, $score(P_2) = 17.43$, and $score(P_3) = 27.00$. All values are then normalized to the interval [0,1], hence $score(P_1) = 0.711$, $score(P_2) = 0.645$, $score(P_3) = 1.000$, and we take $score(CSC_1) = 26/27 = 0.963$ as a benchmark.

Cloud security assessment process with methodology MIP for CSC_2 and CSC_3 is the same as above for CSC_1, thus in the following we only report the results. For CSC_2 we get $score(P_1) = 0.736$, $score(P_2) = 0.573$, $score(P_3) = 1.000$, and $score(CSC_2) = 0.963$ as a benchmark. For CSC_3 we obtain $score(P_1) = 0.687$, $score(P_2) = 0.702$, $score(P_3) = 1.000$, and $score(CSC_3) = 0.963$ as a baseline.

As reported in Luna et al. (2015), with the QHP the priority vectors PV_k for CSC_k, $k = 1,2,3$, are as follows:

$PV_1 = [0.2307, 0.2257, 0.2718, 0.2718]$

$PV_2 = [0.2412, 0.2208, 0.2690, 0.2690]$

$PV_3 = [0.2253, 0.2269, 0.2813, 0.2813]$

To compare the results obtained by the QHP and the MIP, we normalize the QHP results to the interval [0,1]. A side by side comparison is shown in Fig. 8.

For all three CSCs there is only one CSP that meets all expressed requirements, namely CSP P_3. CSPs P_1 and P_2 can provide the requested security levels for some SLOs, but not all. Therefore their scores are lower than the benchmark.

The first difference between the results obtained with the QHP and the MIP is that with the MIP the CSC is able to see whether a CSP is able to provide more in terms of security as what was asked for, whereas with the QHP over-provisions are not visible. In graphs in Fig. 8 this is seen in case of P_3 that is able to provide all security assurances required by the CSC. Moreover, for SLO *RI*1.1 CSP P_3 is able to also provide one higher security level which is seen with the MIP but not with the QHP.

The second difference between the MIP and the QHP is in the way they evaluate CSPs that do not fulfill with the CSC requirements.

For the CSC_1 the QHP outlines only a slight difference between P_1 and P_2 although P_1 is able to provide required values for four SLOs (*CO*1.2, *CO*3.1, *FS*1.2, and *FS*2.2) for which P_2 under-provisions, whereas P_2 is able to provide required values for two SLOs (*CO*2.1 and *FS*1.1) for which P_1 under-provisions. Moreover, for the SLO *CO*3.3 where neither is able to fulfill with CSC request, offer from P_1 is closer to the required value than P_2.

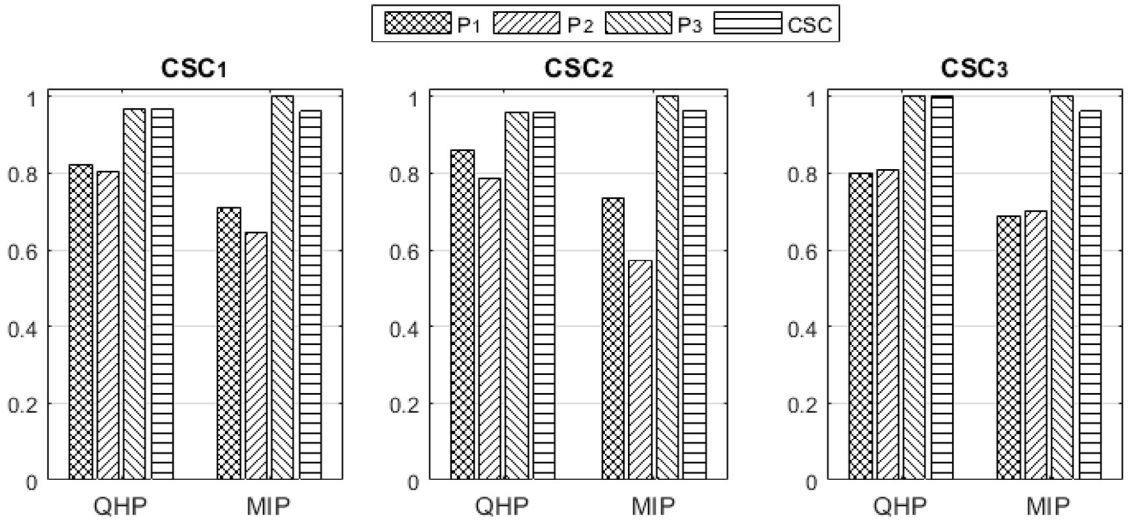

Fig. 8 – Comparison of QHP and MIP.

Hence, the difference between scores for P_1 and P_2 for CSC_1 should be bigger (as outlined by the MIP).

For CSC_2 the difference between the scores for P_1 and P_2 should be even bigger (as identified by the MIP) considering that P_1 is able to grant higher security levels for three high important SLOs ($CO1.2$, $FS1.2$, and $FS2.2$) and one medium important SLO $CO3.3$, whereas P_1 is better for one SLO, namely $CO2.1$. Note that the SLO $CO2.1$, where P_2 assures better security, is ignored in the assessment process because the CSC labeled is as low important (the SLO that are low important to the CSC are assigned weight 0).

In case of CSC_3, both methodologies produce similar scores. On one hand, P_1 ensures better security for two SLOs ($CO1.2$ and $CO3.3$) with providing one security level more than P_2. On the other hand, P_2 meets requirements for 2 SLOs (as opposed to P_1) and grants 4 more security levels versus P_1. Hence P_2 is assigned a slightly better score than P_1.

In summary, by taking into consideration the entire range of security values offered by CSPs versus simple normalization to CSC needs as in the QHP, the MIP methodology results in better overall assessment of Cloud security providers.

7. Conclusions

In this paper, we have developed and validated two efficient Cloud security assessment methodologies that can be used in real-time assessment where efficiency is the key. By decreasing the time required to rank CSPs according to the CSCs requirements, the CSCs are able to adjust their requests and perform assessment dynamically.

We have simplified one of the existing assessment techniques, namely the QHP (Taha et al., 2014), by applying some basic mathematical principles, and we have improved its performance to the point where the derived fQHP methodology can be used in real-time. Additionally, we have proposed a new assessment methodology, namely the MIP, that not only shows better performance with respect to the fQHP, but taking into account to what extent CSPs are able to over-proivision CSC requirements, also shows higher accuracy.

In our future work, we intend to integrate in the assessment process the cost related trade-offs, we want to enable the CSCs to have the opportunity to assign not only one but a range or a set of desired values for each SLO in the SecSLA, and even take into account that some CSPs, when providing a certain level of security for an SLO, are not able or willing to provide also all lower levels.

Acknowledgements

The research presented in this paper was supported, in part, by grants FP7-ICT-2013-11610795 (SPECS) and H2020-644579 (ESCUDO-CLOUD).

REFERENCES

Alabool H, Mahmood A. Trust-based service selection in public cloud computing using fuzzy modified VIKOR method. Aust J Basic Appl Sci 2013;7(9):211–20.

Almorsy M, Grundy J, Ibrahim AS. Collaboration-based cloud computing security management framework. In: Proceedings of the 2011 IEEE 4th international conference on cloud computing, CLOUD'11. Washington, DC: IEEE Computer Society; 2011. p. 364–71.

ASP Industry Consortium, White paper on service level agreements. http://presseservice.pressrelations.de/standard/dereferrer.cfm?r=43129; 2000.

Casola V, Preziosi R, Rak M, Troiano L. A reference model for security level evaluation: policy and fuzzy techniques. J Univers Comput Sci 2005;11(1):150–74.

Casola V, Mazzeo A, Mazzocca N, Rak M. A SLA evaluation methodology in service oriented architectures. In: Gollmann D, Massacci F, Yautsiukhin A, editors. Quality of protection security measurements and metrics. New York, NY, USA: Springer US; 2006. p. 119–30.

Casola V, Mazzeo A, Mazzocca N, Vittorini V. A policy-based methodology for security evaluation: a security metric for public key infrastructures. J Comput Sec 2007;15(2):197–229.
Cloud Security Alliance (CSA), Consensus assessments initiative questionnaire v1.1. https://cloudsecurityalliance.org/download/consensus-assessments-initiative-questionnaire-v1-1/; 2011.
Cloud Security Alliance (CSA), Cloud controls matrix v3. https://cloudsecurityalliance.org/research/ccm/; 2014a.
Cloud Security Alliance (CSA), Consensus assessments initiative questionnaire v3.0.1. https://cloudsecurityalliance.org/download/consensus-assessments-initiative-questionnaire-v3-0-1/; 2014b.
Cloud Security Alliance (CSA), Security, trust and assurance registry (star). https://cloudsecurityalliance.org/star/; 2015a.
Cloud Security Alliance (CSA), Open certification working group. https://cloudsecurityalliance.org/group/open-certification/; 2015b.
Dekker M, Hogben G. Survey and analysis of security parameters in cloud SLAs across the European public sector. http://www.enisa.europa.eu/activities/Resilience-and-CIIP/cloud-computing/survey-and-analysis-of-security-parameters-in-cloud-slas-across-the-european-public-sector; 2011.
Demmel JW. Applied numerical linear algebra. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1997.
Diver S. Information security policy – a development guide for large and small companies. https://www.sans.org/reading-room/whitepapers/policyissues/information-security-policy-development-guide-large-small-companies-1331; 2007.
European Commission (EC), Cloud service level agreement standardisation guidelines (C-SIG SLA 2014). 2014.
European Telecommunications and Standards Institute (ETSI), Cloud standards coordination final report. http://csc.etsi.org/resources/CSC-Phase-1/CSC-Deliverable-008-Final_Report-V1_0.pdf; 2013.
Frankova G, Yautsiukhin A. Service and protection level agreements for business processes. In: Proceedings of the 2nd European young researchers workshop on service oriented computing, YRSOC'07. Leicester, UK: University of Leicester; 2007. p. 38–43.
Garg SK, Versteeg S, Buyya R. SMICloud: a framework for comparing and ranking cloud services. In: Proceedings of the 2011 fourth IEEE international conference on utility and cloud computing, UCC '11. Washington, DC: IEEE Computer Society; 2011. p. 210–18.
Habib SM, Ries S, Muhlhauser M. Towards a trust management system for cloud computing. In: Proceedings of the 2011IEEE 10th international conference on trust, security and privacy in computing and communications, TRUSTCOM '11. Washington, DC: IEEE Computer Society; 2011. p. 933–9.
International Organization for Standardization (ISO/IEC), ISO/IEC 27002:2013 Information technology – Security techniques – Code of practice for information security controls (Second edition). http://www.iso27001security.com/html/27002.html; 2013.
International Organization for Standardization (ISO/IEC), ISO/IEC 19086, Information Technology – cloud computing – Service level agreement (SLA) framework and terminology (Draft). 2014.
Krautsevich L, Martinelli F, Yautsiukhin A. A general method for assessment of security in complex services. In: Proceedings of the 4th European conference on towards a service-based internet, ServiceWave'11. Berlin, Heidelberg: Springer-Verlag; 2011. p. 153–64.
Lewis L. Managing business and service networks. New York, NY: Springer US; 2002.
Li A, Yang X, Kandula S, Zhang M. CloudCmp: comparing public cloud providers. In: Proceedings of the 10th ACM SIGCOMM conference on internet measurement, IMC '10. New York, NY: ACM; 2010. p. 1–14.
Ludwig H, Keller A, Dan A, King RP, Franck R. Web service level agreement (WSLA) language specification. http://www.research.ibm.com/people/a/akeller/Data/WSLASpecV1-20030128.pdf; 2002.
Luna J, Ghani H, Germanus D, Suri N. A security metrics framework for the Cloud. In: Proceedings of the 2011 IEEE international conference on security and cryptography, SECRYPT'11. Washington, DC: IEEE Computer Society; 2011. p. 245–50.
Luna J, Langenberg R, Suri N. Benchmarking cloud security level agreements using quantitative policy trees. In: Proceedings of the 2012 ACM workshop on cloud computing security workshop, CCSW '12. New York, NY: ACM; 2012. p. 103–12.
Luna J, Taha A, Trapero R, Suri N. Quantitative reasoning about cloud security using service level agreements. IEEE Trans Cloud Comput 2015;doi:10.1109/TCC.2015.2469659.
Menzel M, Ranjan R. CloudGenius: decision support for web server cloud migration. In: Proceedings of the 21st international conference on world wide web, WWW '12. New York, NY, USA: ACM; 2012. p. 979–88.
National Institute of Standards and Technology (NIST), Cloud computing: cloud service metrics description (Draft). http://www.nist.gov/itl/cloud/upload/RATAX-CloudServiceMetricsDescription-DRAFT-20141111.pdf; 2014.
Noor TH, Sheng QZ. Trust as a service: a framework for trust management in cloud environments. In: Proceedings of the 12th international conference on web information system engineering, WISE'11. Berlin, Heidelberg: Springer-Verlag; 2011. p. 314–21.
Rehman ZU, Hussain FK, Hussain OK. Towards multi-criteria cloud service selection. In: Proceedings of the 2011 fifth international conference on innovative mobile and internet services in ubiquitous computing, IMIS '11. Washington, DC: IEEE Computer Society; 2011. p. 44–8.
Rehman ZU, Hussain OK, Hussain FK. Iaas cloud selection using MCDM methods. In: Proceedings of the 2012 IEEE ninth international conference on e-business engineering, ICEBE '12. Washington, DC: IEEE Computer Society; 2012. p. 246–51.
Swanson M, Hash J, Bowen P. NIST special publication 800-18 Guide for developing security plans for federal information systems. http://csrc.nist.gov/publications/nistpubs/800-18-Rev1/sp800-18-Rev1-final.pdf; 2005
Taha A, Metzler P, Trapero R, Luna J, Suri N. Identifying and Utilizing Dependencies Across Cloud Security Services. In: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security (*ASIA CCS '16*). New York, NY, USA: ACM; 2016. p. 329–40.
Taha A, Trapero R, Luna J, Suri N. AHP-based quantitative approach for assessing and comparing cloud security. In: Proceedings of the 2014 IEEE 13th international conference on trust, security and privacy in computing and communications, TRUSTCOM '14. Washington, DC: IEEE Computer Society; 2014. p. 284–91.
White K. Definition of managed objects for service level agreements performance monitoring, RFC 2758, IETF. https://tools.ietf.org/html/rfc2758; 2000.

Jolanda Modic received her PhD in 2014 from the Faculty of Mathematics and Physics, University of Ljubljana. She is currently a researcher at XLAB, Ljubljana, Slovenia, and is focused on automated SLA management frameworks, and evaluating, monitoring, and enforcing security in cloud environments.

Ruben Trapero received his PhD from Universidad Politécnica of Madrid and was an assistant professor at Universidad Carlos III of

Madrid. Since 2014 he is a lead researcher at the Technische Universität of Darmstadt, Germany. His research interests include privacy, identity management, cloud security and service engineering.

Ahmed Taha is currently a PhD student at DEEDS group in the Department of Computer Science at Technische Universität of Darmstadt, Germany. His research interest includes Cloud security metrics, security assessment and quantification.

Jesus Luna is the Research Director of the Cloud Security Alliance (Europe). His main responsibilities include the internal scientific/technical management of CSA's funded projects. Jesus obtained his PhD degree in Computer Architecture from the Technical University of Catalonia (2008). Since 2003, Jesus is also affiliated with the Technische Universität of Darmstadt, Germany. His main research interests are security quantification, cloud security, and security policies.

Miha Stopar received his BSc in 2007 from the Faculty of Mathematics and Physics, University of Ljubljana. He is currently a researcher and a software developer at XLAB, Ljubljana, Slovenia. His research interests include cloud computing, machine learning, and applied cryptography.

Neeraj Suri received his PhD from the University of Massachusetts at Amherst and is a Chair Professor at the Technische Universität of Darmstadt, Germany. His research addresses the design, analysis, and assessment of trustworthy cloud services.