Multi-dimensional Secure Service Orchestration

Gabriele Costa[1], Fabio Martinelli[2(✉)], and Artsiom Yautsiukhin[2]

[1] Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi, Universitá di Genova, Genova, Italy
gabriele.costa@unige.it

[2] Istituto di Informatica e Telematica Consiglio Nazionale delle Ricerche, Pisa, Italy
{fabio.martinelli,artsiom.yautsiukhin}@iit.cnr.it

Abstract. Web services composition allows a software designer for combining atomic services, for instance taken from a marketplace, in a complex business process fulfilling a desired functional goal. Moreover, among a large number of possible compositions, the designer may want to consider only those which satisfy specific non-functional requirements.

In our work we consider verification of security properties and evaluation quantitative security metrics in a single framework. The main focus of this article is the verification of a composition with several security metrics at once. We provide a general solution for the problem and show how such verification can be made more efficient in specific cases (e.g., when a metric is an abstraction of another one). We employ a mathematical structure called c-semirings granting the generality of our approach.

1 Introduction

Service composition allows a service designer to create a new complex service out of a set of available services (announced in a Marketplace). Often the result of such process is a set of alternative compositions, which fulfil the same functional goal but have different Quality of Service (QoS). Providers of complex services want to obtain the highest quality services and to guarantee this quality even if some problems with components arise. Naturally, security is one of such qualities.

A number of techniques were provided to obtain the evidences whether the service composition satisfies some security properties [1–5]. Many of these techniques use formal methods to model a complex service and to proof the compliance of this model with a security specification. First, it is important to model services in a "safe" way in order not to miss any security-relevant behaviour. Secondly, the actual service implementation must comply with its specification to assure that the results of the analysis are valid.

Some security properties are of quantitative nature and a decision about whether they are satisfied depends on the concrete requirements of the customer [6]. Thus, when a service provider advertises the QoS of its service it

This work was partly supported by EU-FP7-ICT NESSoS, EU-FP7-ICT ANIKETOS and EU-FP7-ICT SPaCIoS projects.

needs to specify the values of security metrics for the service. Some mathematical models were proposed to analyse service composition with security metric [7,8].

Since, there is no one reliable security metric which completely and unambiguously describes the security level provided by a service [9,10] several quantitative security requirements could be considered by a service orchestrator at once [8,11]. Moreover, the same security qualities may be expressed in a slightly different manner. Therefore, there is a need for a framework which not only evaluates a service composition using several metrics, but also works with different types of similar metrics.

In this paper we extend our previous work [12,13] (based on a type and effect system of Bartoletti et al. [3]) on secure service orchestration in which we provided a single framework for analysis of security properties and security metrics. In this article we show how multi-dimensional security metrics can be incorporated into our framework without violating a safety property. We apply n-dimensional c-semirings to preserve generality of our approach. Although we may not always choose the best/worst option, we show that metric abstraction can help to do this in some cases. In this paper we focus on security metrics but the approach may be generalised for other quantitative qualities of services.

This paper is organised as follows. We start with a running example (Sect. 2). Then, we recall some features from our previous work (Sect. 3). Section 4 contains the core ideas on aggregation of different security metrics. We finish the paper with related work (Sect. 5) and conclusions (Sect. 6).

2 Running Example

A travel agency BestTravel, which offers a travel planning service, moves a part of its business to the web. BestTravel exploits existing services for implementing its process, which includes three sub-processes: (i) find_a_connection, (ii) find_a_hotel, (iii) prepare_invoice. A service developer starts with creation of an abstract workflow, which defines the general process but does not assign concrete services to the defined tasks (e.g., see Fig. 1 represented using BPMN [14]).

Reading Fig. 1 (from left to right) a process of BestTravel works as follows. First, BestTravel finds_a_connection and finds_a_hotel in parallel (rooted in $\oplus$). The find_a_connection sub-process consists of `searching_for_a_direct_flight` and `booking_the_direct_flight`. If the cheap direct flight was not found the service `searches_for_an_itinerary` and `books_the_itinerary`. An itinerary also may be a direct flight but it costs more than the direct option considered before. In parallel BestTravel `searches_for_a_hotel` and `books_the_hotel`. Finally, BestTravel `signs_the_receipt`.

There are 10 concrete services found in a marketplace suitable for the defined tasks. Table 1 displays these concrete services and their mapping to the abstract services. All services specify values of several security-relevant parameters. Note, that sometimes parameters are of different kind. For example, trust value for Windjet and Ryanair are discrete values from a set {1, 2, 3, 4, 5}, when other

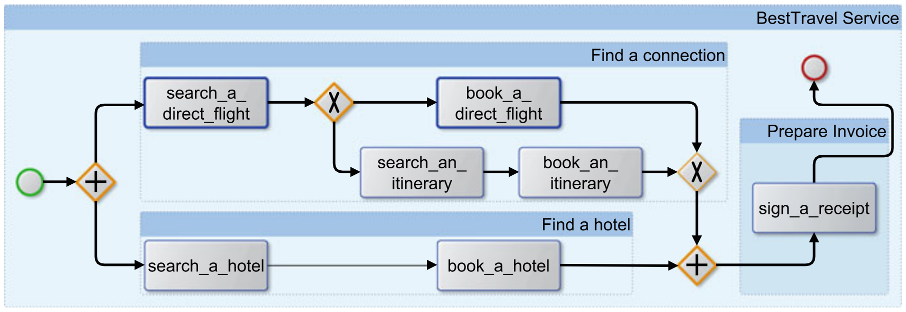

Fig. 1. Abstract workflow for BestTravel.

Table 1. Abstract and concrete services

Abstract	Index	Concrete	Risk	Trust	Recovery time
Search a direct flight	1	Windjet	10	5	75
	2	Ryanair	20	4	45
Search an itinerary	3	Lufthansa	5	0.98	fast
	4	Airfrance	8	0,95	normal
Booking service	5	Paypal	5	0.95	30
	6	Ripplepay	15	0.89	60
Search a hotel	7	HotelBooker	40	0.93	60
	8	HotelClub	30	0.92	90
Sign a receipt	9	ESignForms	0.3; 0.6; 0.9	0.73	150
	10	VeriSign	0.4; 0.5; 0.8	0.87	200

trust values are from [0;1] interval. In contrast to other services, Lufthansa and AirFrance express the recovery time as qualitative values. Finally, risk values of ESignForms and VeriSign represented as triples containing the probability of not violating integrity, confidentiality and availability.

BestTravel has several security requirements for the created process. First, BestTravel wants to have *risk* level of find_a_flight and find_a_hotel reservation sub-processes less than *75* euro (measured as Annualised Loss Expectancy (ALE) [15]). Furthermore, the two sub-processes must have the overall *trust* rating not lower than *0.8* (or not lower than 5 for a discrete scale). The *time of recovery* of the sub-processes should not be more than *120* min. Finally, the risk value for `sign_a_receipt` must be *medium* or smaller.

3 Background

3.1 C-Semirings

We exploits the notion of *c-semiring* [16] for the abstraction of metrics and operators over metrics to provide a generic framework for all metrics which could be considered as c-semirings. A c-semiring consists of a domain of values D, and two types of operators: multiplication ($\otimes$) and addition ($\oplus$). Formally, a c-semiring is defined as follows (see Bistarelli et al., [16] for details).

Definition 1. *A* c-semiring *$\mathbb{S}$ is a tuple $\langle D, \oplus, \otimes, \mathbf{0}, \mathbf{1}\rangle$ where*

- *D is a (possibly infinite) set of elements and $\mathbf{0}$, $\mathbf{1} \in D$;*
- *$\oplus$, being an addition defined over D, is a binary, commutative, idempotent, and associative operator, such that $\mathbf{0}$ is its* unit *element and $\mathbf{1}$ is its absorbing element;*
- *$\otimes$, being a multiplication over D, is a binary, commutative, associative, and distributive over addition operator, such that $\mathbf{1}$ is its* unit *element and $\mathbf{0}$ is its* absorbing *element;*

Definition 2. *$\leq_S$ is a partial order relation over D: $d_1 \leq_S d_2$ iff $d_1 \oplus d_2 = d_2$.*

In this work we need a reverse operation, which returns the worst possible value for summation $\oplus^{-1}$ which is defined as follows.

Definition 3. $d_1 \oplus^{-1} d_2 = glb(d_1, d_2)$;

where $glb(d_1, d_2) = d$ if and only if (i) $d \leq_S d_1 \wedge d \leq_S d_2$, and (ii) $\forall\ d'\ .\ d' \leq_S d_1 \wedge\ d' \leq_S d_2 \rightarrow\ d' \leq_S d$

The definition of the reverse operation returns the opposite value of direct addition operation when the operation is defined. For the cases in which the addition operation is undefined, the greatest lower bound (glb) is returned[1].

Property 1. Operation $\oplus^{-1}$ is associative, commutative, idempotent, distributive over $\otimes$, monotone[2].

Example 1. Regarding to the security targets BestTravel is going to use three metrics: trust, risk, and recovery time. Trust is often seen as a probability that a provider behaves according to the contract, i.e., for majority of services in the marketplace (see Table 1) trust has a value between 0 and 1. The values are aggregated by multiplication, and the highest values is preferable. Thus, c-semiring for trust formally is defined as follows: $\mathtt{S}^1 = \langle [0, 1], max, \times, 0, 1\rangle$.

Risk is often considered as possible losses and has the domain of positive natural numbers. Aggregation of risk values is summation of losses, and a lower risk is better than a higher one. Thus, c-semiring for risk is $\mathtt{S}^2 = \langle \mathbb{N}^+ \cup \{\infty\}, min, +, \infty, 0\rangle$.

Recovery time denotes the time required for the service to recover after an incident (e.g., after a successful DOS attack). For majority of services in the marketplace (see Table 1) time has domain of positive real numbers. Aggregation of recovery time values is a maximising operation, while a lower time of recovery is considered better than a higher one. Thus, c-semiring for recovery time is $\mathtt{S}^3 = \langle \mathbb{N}^+ \cup \{\infty\}, min, max, \infty, 0\rangle$.

[1] Note that the existence of glb is granted by the presence of top element in our domains.

[2] A link with proofs: http://wwwold.iit.cnr.it/staff/artsiom.yautsiukhin/Resources/Proofs-SBP.pdf.

Table 2. Syntax of history expressions.

$$H, H' ::= \varepsilon \mid h \mid \alpha(r) \mid H \cdot H' \mid H + H' \mid H \mid H' \mid \bar{d}\#H \mid \varphi[H] \mid \gamma\langle H\rangle \mid \mu h.H$$

$$H_{BT} = \left(\gamma \left\langle (H_1 + H_2) \cdot \left(\begin{array}{c} (H_5 + H_6) \\ + \\ ((H_3 + H_4) \cdot (H_5 + H_6)) \end{array} \right) \right\rangle \middle| \gamma \left\langle \begin{array}{c} (H_5 + H_6) \\ \cdot \\ (H_7 + H_8) \end{array} \right\rangle \right) \\ \cdot \gamma \langle \mu h.((H_9 + H_{10}) \cdot h + \varepsilon)\rangle$$

Fig. 2. History expression for BestTravel.

3.2 History Expressions and Services

In our previous paper [12] we have described how history expressions, originally proposed to model the behaviour of complex services [17], can be suitably enriched with metric annotations. Also, History expressions can be adopted for verification of temporal properties by checking whether they satisfy a corresponding specification (e.g., a LTL formula or a security automaton). History expressions can be inferred from service implementation by means of a suitable type and effect system which grants the soundness, namely *type safety*, of the resulting expressions. Here we do not detail the type and effect inference process and we refer the interested reader to [12] and to [17].

In this paper we assume the history expression inference process for service implementation as given. We only recall the inference rule for access events below. Also, we (informally) recall the type safety theorem stated in [12].

Theorem 1. *If the type and effect system infers a history expression H from a service e, then every possible execution trace of e is denoted by H.*

The syntax of the history expression is shown in Table 2. Intuitively, history expressions can be empty (i.e. ε), variables (ranged over by h, h') or access operations (access α to a resource r, in symbols $\alpha(r)$). Also, a history expression can be a sequence $H \cdot H'$, a choice $H + H'$, a parallel composition $H \mid H'$ or a recursion $\mu h.H$. Finally, history expressions can be annotated with metric vectors, e.g., $\bar{d}\#H$, security framings, e.g., $\varphi[H]$, and metric framings, e.g., $\gamma\langle H\rangle$. Their meaning is straightforward. The annotation[3] $\bar{d}\#H$ says that (the service associated to the history expression) H can originate metric vectors which are bounded by $\bar{d}$. A security framing $\varphi[H]$ says that, over the execution histories produced by H, the security policy φ holds. Similarly, $\gamma\langle H\rangle$ applies a metric policy γ to H.

[3] Here we go ahead a bit and use vectors of values instead of simple values. In the following we show that such substitution is just.

Example 2. Using the type and effect inference presented in [12], we associate the history expression of Fig. 2 to the BestTravel service (see Fig. 1) and in Example 3 we will add metric values to these expressions.

We proposed to aggregate metrics (specified as c-semirings) according to the business process of complex web services using equational rules (see Table 3). In short, security and trust metrics are expressed as c-semirings and assigned to specific history expressions (i.e., $\bar{d}\#H$, called metric normal form (MNF)) in the history expression for the composite service (similar to Fig. 2). Then the equational rules are consequently applied to find the values of metrics for the complete process. In fact, Table 3 states, that multiplication operator is used for aggregation of values for parallel ($\bar{d}_1\#H_1 \mid \bar{d}_2\#H_2$), sequential ($\bar{d}_1\#H_1 \cdot \bar{d}_2\#H_2$) and cyclic ($\mu h.H$) execution of services, while reverse addition is used for non-deterministic choice ($\bar{d}_1\#H_1 + \bar{d}_2\#H_2$) and selection of the worst alternative [12]. Such an aggregation process results can be used to advertise the level of security provided by the composite web service. Also, in [12] we proved that $\equiv$ is an equivalence relation for history expressions semantics. Hence, although security is not a compositional property in general, security analysis can be safely carried out on the aggregation of history expressions.

Table 3. Equational rules.

$$H \equiv \mathbf{1}\#H \quad \bar{d}_1\#\bar{d}_2\#H \equiv \bar{d}_2\#\bar{d}_1\#H \equiv \bar{d}_1 \otimes \bar{d}_2\#H$$

$$\bar{d}_1\#H_1 \cdot \bar{d}_2\#H_2 \equiv \bar{d}_1 \otimes \bar{d}_2\#(H_1 \cdot H_2) \quad \varphi[\bar{d}\#H] \equiv \bar{d}\#\varphi[H]$$

$$\bar{d}_1\#H_1 + \bar{d}_2\#H_2 \equiv \bar{d}_1 \oplus^{-1} \bar{d}_2\#(H_1 + H_2) \quad \bar{d}_1\#H_1 \mid \bar{d}_2\#H_2 \equiv \bar{d}_1 \otimes \bar{d}_2\#(H_1 \mid H_2)$$

$$\gamma\left\langle \bar{d}\#H \right\rangle \equiv \bar{d}'\#\gamma\left\langle H \right\rangle \quad \text{where } \gamma = T \geq_T \bar{d}'' \text{ and } \bar{d}' = \bar{d} \oplus^{-1} \bar{d}''$$

$$\mu h.H \equiv \bar{d}''\#\mu h.H' \quad \text{where } \bar{d}'' = \bigoplus_n{}^{-1} \Phi^n(\mathbf{0}) \text{ and } \Phi(\bar{d}) = \bar{d}' \Leftrightarrow \begin{cases} H[\bar{d}\#h/h] \equiv \bar{d}'\#H' \\ \wedge \\ \bar{d}'\#H' \text{ is in MNF} \end{cases}$$

4 Aggregation of Several Security Metrics with C-Semirings

In practice, several security parameters are required to assess a service. In this case we should use an n-dimensional c-semiring [18]:

Definition 4. *Assume that we have n c-semirings $\boldsymbol{S}^i$, $0 < i \leq n$ (we also add upper index i to every parameter of a semiring, i.e., $\boldsymbol{S}^i = \langle D^i, \oplus^i, \otimes^i, \boldsymbol{0}^i, \boldsymbol{1}^i \rangle$). An n-dimensional c-semiring is $\bar{\boldsymbol{S}} = \langle \bar{D}, \oplus, \otimes, \bar{\boldsymbol{0}}, \bar{\boldsymbol{1}} \rangle$, where $\bar{D} = (D^1, ..., D^n)$, including $\bar{\boldsymbol{0}} = (\boldsymbol{0}^1, ..., \boldsymbol{0}^n)$ and $\bar{\boldsymbol{1}} = (\boldsymbol{1}^1, ..., \boldsymbol{1}^n)$. For any two vectors of values $\bar{d}_1$ and $\bar{d}_2$ of $\bar{\boldsymbol{S}}$ the multiplication operations is defined as follows: $\bar{d}_1 \otimes \bar{d}_2 = (d_1^1 \otimes^1 d_2^1, ..., d_1^n \otimes^n d_2^n)$. The additional operation is defined using Pareto-optimality: $\bar{d}_1 \oplus \bar{d}_2 = \bar{d}_2 \; iff \; \forall i \; d_1^i \oplus^i d_2^i = d_2^i$.*

Table 4. Metric annotation of concrete sub-services.

$H_1 = \bar{d}_1 \# H'_1 = (10, 5, 75) \# H'_1$	$H_6 = \bar{d}_6 \# H'_6 = (15, 0.89, 60) \# H'_6$
$H_2 = \bar{d}_2 \# H'_2 = (20, 4, 45) \# H'_2$	$H_7 = \bar{d}_7 \# H'_7 = (40, 0.93, 60) \# H'_7$
$H_3 = \bar{d}_3 \# H'_3 = (5, 0.98, \text{fast}) \# H'_3$	$H_8 = \bar{d}_8 \# H'_8 = (30, 0.92, 90) \# H'_8$
$H_4 = \bar{d}_4 \# H'_4 = (8, 0.95, \text{normal}) \# H'_4$	$H_9 = \bar{d}_9 \# H'_9 = ((0.3; 0.6; 0.9), 0.73, 150) \# H'_9$
$H_5 = \bar{d}_5 \# H'_5 = (5, 0.95, 30) \# H'_5$	$H_{10} = \bar{d}_{10} \# H'_{10} = ((0.4; 0.5; 0.8), 0.87, 200) \# H'_{10}$

In the article we use upper indexes to denote different c-semirings, which compose an n-dimensional c-semiring, while the lower indexes denote different instances of a c-semiring. For the sake of presentation, we also specify an n-dimensional c-semiring as a vector of c-semirings, e.g., $\bar{\mathtt{S}} = (\mathtt{S}^1, \mathtt{S}^2, ..., \mathtt{S}^n)$. Note, that n-dimensional c-semiring is a c-semiring as well [16]. This means that all our formulas written for the quantitative analysis of a composite service presented in [12] are relevant for the new c-semiring structure.

A crucial property we want to prove is that usage of n-dimensional c-semirings does not invalidate the semantics of history expressions, i.e., it does not affect the safety property stated by Theorem 1.

Property 2. For all history expressions H and H' if $H \equiv H'$ then each execution trace is denoted by H if and only if it is also denoted by H'.

In general, we know that all history expressions can be reduced to a corresponding MNF as stated by the following property.

Property 3. For each history expression H there exists H' such that $H \equiv H'$ and H' is in MNF.

The last property we recall from [12] is *metric safety*, which characterises one of the main aspects of the metric annotations we generate.

Theorem 2. *If the type and effects system infers H from a service e and $H \equiv \bar{d} \# H'$ such that $\bar{d} \# H'$ is in MNF, then for each execution of e starting from vector $\bar{d}'$ and generating $\bar{d}''$ holds that $\bar{d}'' \leq_S \bar{d}' \otimes \bar{d}$.*

Similarly to type safety, this theorem guarantees that metric annotations produced by our equational theory provide an upper bound to the metric values generated by the execution of a term. As each of them has a corresponding MNF, this theorem can be universally applied to any history expression. Referring to our working example, the theorem above guarantees that we can always build a table like Table 1 starting from the implementation of the involved services (see [12] for details about the automatic assignment of metric annotations to history expressions).

Example 3. In Fig. 2, H_{BT} denotes the behaviour of BestTravel. In particular, we use H_i as an abbreviation for the history expression of sub-service i and $\bar{d}_i$ for the annotating metric. The structure of each history expression H_i is reported in Table 4. Each history expression H_i has the form $\bar{d}_i \# H'_i$ where H'_i has no further metric annotations, that is, H_i is in *metric normal form.*

Example 4. In our running example the n-dimensional c-semiring for the find_a_hotel sub-process, can be specified as follows: $\bar{\mathtt{S}} = (\mathtt{S}^1, \mathtt{S}^2, \mathtt{S}^3)$ (see Example 1) We have two alternatives for `search_a_hotel` (see Table 1): $(40, 0.93, 60)\#H'_7 + (30, 0.92, 90)\#H'_8$, and two alternatives for `book_a_hotel`: $(5, 0.95, 30)\#H'_5 + (15, 0.89, 60)\#H'_6$. We see that $\bar{d}_5 \oplus^{-1} \bar{d}_6 = \bar{d}_6$ and these values ($\bar{d}_6$) the BestTravel is able to guarantee if at least one of the two services is available. Unfortunately, $\bar{d}_7 \oplus^{-1} \bar{d}_8$ cannot be solved, since $40 > 30$, but $0.93 > 0.92$ and $60 < 90$ and we have to propagate the glb for `search_a_hotel` activity further. The whole find_a_hotel sub-process has (H_{f_h}) the result $(55, 0.8188, 90)\#H'_{f\_h}$. Note, that the result satisfies0 the requirements specified in Sect. 2.

4.1 Aggregation of Similar Metrics

Sometimes the same property is measured in different ways. For example, security risk level is measured by quantitative (e.g., using natural numbers) and qualitative (e.g., using high, medium, and low levels) methods. Also trust may be computed either as a values in [0;1] interval (similar to eBay reputation system) or as a discrete value (e.g., {-1, 0, 1, 2, 3, 4} [19]).

In order to make our analysis work with different types of metrics we first need to link a more concrete metric and a more abstract one. Such a link must satisfy the conditions for *Galois insertion* to correctly approximate a concrete metric in abstract domain and vice versa (see [16] for details):

Definition 5. *Let we have two sets D^c and D^a and two operations $\sqsubseteq_S$ and $\leq_S$ which define the order in the two sets correspondingly (we write (D^c,$\sqsubseteq_S$) and (D^a,$\leq_S$) to denote these posets). A Galois insertion $\langle\alpha, \gamma\rangle : (D^c, \sqsubseteq_S) \rightleftharpoons (D^a, \leq_S)$ is a pair of mapping functions: $\alpha : D^c \to D^a$ and $\gamma : D^a \to D^c$ such that:*

1. *α and γ are monotone,*
2. *$\forall d^c \in D^c,\ d^c \sqsubseteq_S \gamma(\alpha(d^c))$, and*
3. *$\forall d^a \in D^a,\ \alpha(\gamma(d^a)) = d^a$*

If we can prove that α abstraction satisfies the order-preserving property [18], we can transform all concrete metrics to abstract ones and find the optimal solutions using only one (abstract) set of metrics (see [18] for the proof). Note, that in this case the optimal solution for a more abstract metric may be referred to several concrete solutions.

Definition 6. *The abstraction α is order-preserving if for any two sets D_1^c and D_2^c of concrete elements the following observation holds:*

$$\tilde{\bigotimes_{d \in D_1^c}} \alpha(d) \sqsubseteq_S \tilde{\bigotimes_{d \in D_2^c}} \alpha(d) \Longrightarrow \bigotimes_{d \in D_1^c} d \leq_S \bigotimes_{d \in D_2^c} d$$

where $\tilde{\bigotimes}$ and $\bigotimes$ are multiplicative operations for abstract and concrete c-semirings correspondingly.

Example 5. Without loss of generality we consider recovery time metric in isolation. In the marketplace (see Table 1) there are two alternatives for `search_for_an_itinerary` (Lufthansa and AirFrance) which have values *fast* and *medium* from c-semiring $\langle\{very\ fast, fast, normal, slow, very\ slow\}, min, max, very\ slow, very\ fast\rangle$. Note, that `booking_services` (PayPal and Ripplepay) have values 30 and 60 from a different c-semiring $\langle N^{+} \cup \{\infty\}, min, max, \infty, 0\rangle$. The α abstraction in this case is defined as the following mapping: $[0, 15] \mapsto veryfast$, $(15, 50] \mapsto fast$, $(50, 100] \mapsto normal$, $(100, 300] \mapsto slow$, $[300, +\infty] \mapsto veryslow$. The backward transformation γ: $very\ fast \mapsto 15$; $fast \mapsto 50$; $normal \mapsto 100$; $slow \mapsto 300$; $very\ slow \mapsto +\infty$.

Thus, the value of the first option for `booking` activity (PayPal) is mapped to *fast* in the more abstract c-semiring, when the second option (Ripplepay) has *medium* value. To be in a safe position we consider the worst case, getting *medium* value for the sub-process.

Unfortunately, many abstractions do not have such property and we have to use glb for aggregation.

In many cases a service may define a metric only in one c-semiring, while another service defines a similar metric with another c-semiring. In order to aggregate or select a value in such situation we can assign the best value (i.e., $\mathbf{1}$) to the undefined c-semiring without changing the result. We can do this, since $\mathbf{1}$ is a unit element for $\otimes$ and $\oplus^{-1}$, i.e., $\forall d\ \mathbf{1} \otimes d = d$ and $\mathbf{1} \oplus^{-1} d = d$.

Example 6. The trust metric for `search_a_direct_flight` services is of a different c-semiring than others: $\mathtt{S}^4 = \langle\{1, 2, 3, 4, 5\}, max, min, 1, 5\rangle$. The Galois insertions between $\mathtt{S}^4$ and $\mathtt{S}^3$ we used for trust so far (see Example 1) is defined as follows: α: $[0, 0.2) \mapsto 1$; $[0.2, 0.4) \mapsto 2$; $[0.4, 0.6) \mapsto 3$; $[0.6, 0.8) \mapsto 4$; $[0.8, 1] \mapsto 5$; and γ: $1 \mapsto 0$; $2 \mapsto 0.2$; $3 \mapsto 0.4$; $4 \mapsto 0.6$; $5 \mapsto 0.8$.

Since, we cannot make a final choice about the selected candidate for `search_a_direct_flight` subprocess H_{12} we should propagate the glb of them $(10, 5, 75)\#H'_1$ and $(20, 4, 45)\#H'_2$, which is $(20, 4, 75)\#H'_{12}$. The result of the check for the following non-deterministic choice (H_{11}) is equal to $(23, 0.8455, normal)\#H'_{11}$: the branch of booking an itinerary has the worst value (after aggregation of the worst alternatives for `search_an_itinerary` $(8, 0.95, normal)\#H'_4$ and `book_an_itinerary` $(15, 0.89, fast)\#H'_6$). In order to aggregate values of $\bar{d}_{12}$ and $\bar{d}_{11}$ we need to have two dimensions for two types of trust metrics. Thus, our n-dimensional c-semiring transforms to $\bar{\mathtt{S}}' = (\mathtt{S}^1, \mathtt{S}^2, \mathtt{S}^4, \mathtt{S}^3)$. The c-semirings under consideration are transformed to $(20, 1, 4, 75)'\#H'_{12}$, and $(23, 0.8455, 5, normal)'\#H'_{11}$. Now, we can easily aggregate the c-semirings. The result for find_a_flight sub-process ($H_{f\text{-}f}$) is $(43, 0.8455, 4, normal)'\#H'_{f\text{-}f}$.

Finally, at the end of the aggregation process we can use abstraction to a metric common for different components of n-dimensional c-semiring for selecting the worst alternative. Note, that adding $\mathbf{1}$ at a place of an absent component does not change the result of the aggregation at the abstract level. First, such addition does not change the result of aggregation and selection, as shown above. Second, even if no aggregation or selection is needed, abstraction maps **1** in a concrete c-semiring to **1** in an abstract c-semiring (see [18]) and thus, does not affect the result of aggregation on the abstract level.

Example 7. We continue Example 6. Now we can transform the more concrete value to the abstract one using α: $0.8455 \mapsto 5$, and aggregate the values of trust c-semirings S^2 and S^4 using aggregation operation of abstract metric: $min(4,5) = 4$. Thus, the result is $(43, 4, normal)\#H'_{f\_f}$. We see, that this sub-process may violate the policy of having the trust value at least 5. On the other hand, for order-preserving abstractions we may do the backward mapping γ to find the lowest bound for recovery time and, this means, that the final value of the recovery time metric does not violate the policy (see Example 5): $normal \mapsto 100 < 120$.

Finally, the abstraction may be used to compare metrics even if an abstract metric is not a component of the n-dimensional c-semiring at all.

Example 8. Now, consider the prepare_invoice sub-process, which consists of one activity only, i.e., `sign_a_receipt`. For the sake of simplicity, we consider only risk metric, since there are no other requirements for this sub-process.

There are two concrete services with assigned 3-dimensional c-semirings. The metrics of the 3-dimensional c-semiring denote the probability of not violating of integrity, confidentiality and availability, which could be seen as probabilistic semirings: $\mathtt{S}^5 = \langle [0;1]; max; \times; 0; 1\rangle$. We have 2 alternatives with the $(\mathtt{S}^5, \mathtt{S}^5, \mathtt{S}^5)$ c-semirings: $(0.3, 0.6, 0.9)\#H'_9 + (0.7, 0.9, 0.6)\#H'_{10}$. These values cannot be simply compared, but we can use an abstractions to qualitative risk value: for integrity α^{int}:$[0, 0.333) \mapsto high$, $[0.333, 0.666) \mapsto medium$, $[0.666, 1] \mapsto low$; for confidentiality α^{conf}: $[0, 0.666) \mapsto medium$, $[0.6661] \mapsto low$; for availability α^{av}: $[0, 0.666) \mapsto medium$, $[0.666, 1] \mapsto low$. Here we assumed, that integrity has high impact, while confidentiality and availability - medium impact and used the Risk-Level Matrix from [20][4].

Now, we are able to compare the alternatives using a c-semiring for qualitative risk: $\mathtt{S}^r = \langle\{low, medium, high\}, min, max, low, high\rangle$. The results of the abstraction are: $(high, medium, low)^r\#H'_9 + (low, low, medium)^r\#H'_{10}$ $(\bar{d}^r_9, \bar{d}^r_{10} \in (\mathtt{S}^r, \mathtt{S}^r, \mathtt{S}^r))$. We aggregate values for every c-semiring separately and see that the risk value for ESignForms is *high* and violates the requirement.

5 Related Work

Many authors proposed formal languages for specifying and verifying agreements, also called *contracts*, between a service provider and a customer. Some authors [5,21] propose formal languages for defining service contracts. Such languages rely on process algebra-like syntax and exploit automatic verification

[4] In this example, we also assume, that security breaches are independent.

techniques for generating service orchestrations. In [22] Martinelli and Matteucci describe how to synthesise a secure orchestrator, i.e., an agent which drives the interaction between two services respects a certain security policy. History expressions have been applied to model and verify service compositions by Bartoletti et al. [3]. They apply local policies through a security framing operator and find service orchestrations respecting all of them. Although, the proposals described above use contracts for the specification and analysis of history-based service properties, none of them allow for the definition of security metrics and restrictions on them as we do in this work.

Jaeger et al. [23] proposed to aggregate quantitative service qualities taking into account workflow of services. Such metrics as mean cost and mean reputation were considered. Yu et al. [8] extended the application of the ideas of Jaeger at. al. to select a composition with the best values of a considered quantitative metric. The authors also defined a set of aggregation functions for some specific metrics and applied algorithms for solving multidimensional multiple choice knapsack problem to find the best alternative which satisfies the considered constraints. Security specific metrics were taken into account by Massacci and Yautsiukhin [7]. The authors created a directed graph using the workflow of a process and defined the aggregation algorithm for monotonic metrics.

All these methods lack of generality since the algorithms should be changed when a new metric is considered. Moreover, most of these proposals consider a single metric at a time. Yu et al. [8] proposed a specific weighted function in order to compute a single combined metric and then use it for service selection. Massacci and Yautsiukhin also extended their framework to perform the analysis with several metrics using Pareto-optimality principle [11]. Although, our work uses the same principle, we also have shown that we can apply abstraction to combine similar metrics more efficiently and without using weighted functions, which are hard to define precisely. Moreover, our metric analysis is merged in a unique framework with security property checks and can be preformed at the same time. Finally, our framework determines the values of QoS the service developer may guarantee even if all best services are not available at the moment.

6 Conclusion

In the paper we provided a framework which allows for checking several quantitative security requirements at the same time. We have found that few changes are required to extend our framework for multidimensional analysis. We also found that when metrics satisfy the order-preserving property we are able to use only the abstract metric for analysis. In case this property fails, we still are able the lower value we can guarantee. In both cases, the proposed method allows eliminating alternatives from the consideration in one run avoiding unnecessary aggregation of these values, making the analysis more efficient. Naturally, the proposed method depends on the correct definitions of c-semirings and abstraction functions. Since both definitions are metric-specific, it is enough to specify them once and reuse them in any scenario afterwards. As a future work, we consider implementation of our framework and testing it in a real scenario.

References

1. Nielson, H.R., Nielson, F.: A flow-sensitive analysis of privacy properties. In: Proceedings of the CSF-07 (2007)
2. Rossi, S., Macedonio, D.: Information flow security for service compositions. In: Proceedings of the ICUMT-09 (2009)
3. Bartoletti, M., Degano, P., Ferrari, G.L.: Planning and verifying service composition. J. Comput. Secur. **17**(5), 799–837 (2009)
4. Bravetti, M., Lanese, I., Zavattaro, G.: Contract-driven implementation of choreographies. In: Kaklamanis, C., Nielson, F. (eds.) TGC 2008. LNCS, vol. 5474, pp. 1–18. Springer, Heidelberg (2009)
5. Padovani, L.: Contract-directed synthesis of simple orchestrators. In: van Breugel, F., Chechik, M. (eds.) CONCUR 2008. LNCS, vol. 5201, pp. 131–146. Springer, Heidelberg (2008)
6. Karabulut, Y., et al.: Security and trust in it business outsourcing: a manifesto. ENTCS, vol. 179. Elsevier, Amsterdam (2006)
7. Massacci, F., Yautsiukhin, A.: Modelling of quality of protection in outsourced business processes. In: Proceedings of the IAS-07. IEEE (2007)
8. Yu, T., Zhang, Y., Lin, K.J.: Efficient algorithms for web services selection with end-to-end qos constraints. ACM Trans. Web **1**, 1–26 (2007)
9. Krautsevich, L., et al.: Formal approach to security metrics. what does "more secure" mean for you? In: Proceedings of the MESSA-10. ACM Press (2010)
10. Jaquith, A.: Security Metrics: Replacing Fear, Uncertainty, and Doubt. Addison-Wesley, Upper Saddle River (2007)
11. Innerhofer-Oberperfler, F., Massacci, F., Yautsiukhin, A.: Pareto-optimal architecture according to assurance indicators. In: Proceedings of the 13th Nordic Workshop on Secure IT Systems (2008)
12. Costa, G., Martinelli, F., Yautsiukhin, A.: Metric-aware secure service orchestration. In: Proceedings of the ICE-12. EPTCS (2012)
13. Costa, G., Degano, P., Martinelli, F.: Modular plans for secure service composition. J. Comput. Secur. **20**(1), 81–117 (2012)
14. OMG: Business Process Model and Notation (BPMN). version 2.0 edn.
15. Gordon, L.A., Loeb, M.P.: Managing Cybersecurity Resources: A Cost-Benefit Analysis. McGraw Hill, New York (2006)
16. Bistarelli, S., Montanari, U., Rossi, F.: Semiring-based constraint satisfaction and optimization. J. ACM **44**, 201–236 (1997)
17. Bartoletti, M., Degano, P., Ferrari, G.-L., Zunino, R.: Secure service orchestration. In: Aldini, A., Gorrieri, R. (eds.) FOSAD 2006/2007. LNCS, vol. 4677, pp. 24–74. Springer, Heidelberg (2007)
18. Bistarelli, S., Codognet, P., Rossi, F.: Abstracting soft constraints: framework, properties, examples. Artif. Intell. **139**, 175–211 (2002)
19. Abdul-Rahman, A., Hailes, S.: A distributed trust model. In: Proceedings of the NSPW. ACM (1997)
20. Stoneburner, G., Goguen, A., Feringa, A.: Risk management guide for information technology systems. Technical, report 800–30, NIST

21. Bravetti, M., Zavattaro, G.: Towards a unifying theory for choreography conformance and contract compliance. In: Lumpe, M., Vanderperren, W. (eds.) SC 2007. LNCS, vol. 4829, pp. 34–50. Springer, Heidelberg (2007)
22. Martinelli, F., Matteucci, I.: Synthesis of web services orchestrators in a timed setting. In: Dumas, M., Heckel, R. (eds.) WS-FM 2007. LNCS, vol. 4937, pp. 124–138. Springer, Heidelberg (2008)
23. Jaeger, M.C., Rojec-Goldmann, G., Muhl, G.: QoS aggregation in web service compositions. In: Proceedings of the CEC-05 (2005)