Privacy-Enhanced Web Service Composition

Salah-Eddine Tbahriti, Chirine Ghedira, Brahim Medjahed, and Michael Mrissa

Abstract—Data as a Service (DaaS) builds on service-oriented technologies to enable fast access to data resources on the Web. However, this paradigm raises several new privacy concerns that traditional privacy models do not handle. In addition, DaaS composition may reveal privacy-sensitive information. In this paper, we propose a formal privacy model in order to extend DaaS descriptions with privacy capabilities. The privacy model allows a service to define a privacy policy and a set of privacy requirements. We also propose a privacy-preserving DaaS composition approach allowing to verify the compatibility between privacy requirements and policies in DaaS composition. We propose a negotiation mechanism that makes it possible to dynamically reconcile the privacy capabilities of services when incompatibilities arise in a composition. We validate the applicability of our proposal through a prototype implementation and a set of experiments.

Index Terms—Service composition, DaaS services, privacy, negotiation

1 INTRODUCTION

Web services have recently emerged as a popular medium for data publishing and sharing on the Web [18]. Modern enterprises across all spectrums are moving towards a service-oriented architecture by putting their databases behind Web services, thereby providing a well-documented, platform independent and interoperable method of interacting with their data. This new type of services is known as DaaS (Data-as-a-Service) services [33] where services correspond to calls over the data sources. DaaS sits between services-based applications (i.e., SOA-based business process) and an enterprise’s heterogeneous data sources. They shield applications developers from having to directly interact with the various data sources that give access to business objects, thus enabling them to focus on the business logic only. While individual services may provide interesting information/functionality alone, in most cases, users’ queries require the combination of several Web services through service composition. In spite of the large body of research devoted to service composition over the last years [24]), service composition remains a challenging task in particular regarding privacy. In a nutshell, privacy is the right of an entity to determine when, how, and to what extent it will release private information [16]. Privacy relates to numerous domains of life and has raised particular concerns in the medical field, where personal data, increasingly being released for research, can be or have been, subject to several abuses, compromising the privacy of individuals [3].

1.1 e-Epidemiological Scenario

Let us consider the services in Table 1 and the following epidemiologist’s query Q “What are the ages, genders, address, DNA, salaries of patients infected with H1N1; and what are the global weather conditions of the area where these patients reside?”

We proposed in [2] a mediator-based approach to compose DaaSs. The mediator selects, combines and orchestrates the DaaS services (i.e., gets input from one service and uses it to call another one) to answer received queries. It also carries out all the interactions between the composed services (i.e., relays exchanged data among interconnected services in the composition). The result of the composition process is a composition plan which consists of DaaS that must be executed in a particular order depending on their access patterns (i.e., the ordering of their input and output parameters). Thus, Q can be answered by composing the following services S1, S2, S3:

1. S1 is invoked with the patients’ zip_code to get information about the weather_conditions. 
2. S2 is invoked with the patients’ date_of_birth, zip_code and salary of obtained patients. 
3. S3 is invoked with the patients’ zip_code to get information about the weather_conditions.

1.2 Challenges

Two factors exacerbate the problem of privacy in DaaS. First, DaaS services collect and store a large amount of private information about users. Second, DaaS services are able to share this information with other entities. Besides, the emergence of analysis tools makes it easier to analyze and synthesize huge volumes of information, hence increasing the risk of privacy violation [21]. In the following, we use our epidemiological scenario to illustrate the privacy challenges during service composition.
1.3.1 Privacy Model

We propose a compatibility matching algorithm to check privacy compatibility between component services within a composition. The compatibility matching is based on the notion of privacy subsumption and on a cost model. A matching threshold is set up by services to cater for partial and total privacy compatibility.

1.3.2 Privacy-Aware Service Composition

We introduce a novel approach based on negotiation to reach compatibility of concerned services (i.e., services that participate in a composition which are incompatible). We aim at avoiding the empty set response for user queries by allowing a service to adapt its privacy policy without any damaging impact on privacy. Negotiation strategies are specified via state diagrams and negotiation protocol is proposed to reach compatible policy for composition.

1.3.3 Negotiating Privacy in Service Composition

In the case when any composition plan will be incompatible in terms of privacy, we introduce a novel approach based on negotiation to reach compatibility of concerned services (i.e., services that participate in a composition which are incompatible). We aim at avoiding the empty set response for user queries by allowing a service to adapt its privacy policy without any damaging impact on privacy. Negotiation strategies are specified via state diagrams and negotiation protocol is proposed to reach compatible policy for composition.

1.4 Paper Organization

The rest of this paper is organized as follows: In Section 2 we review the composition approach proposed in [2] as part of the PAIRSE project. We present our privacy model in Section 3. We introduce the notion of compatibility between privacy policies and requirements in Section 4. In Section 5 we show how our DaaS composition approach is extended within privacy-preserving mechanism. We present our negotiation model in Section 6 to deal with the issue of privacy incompatibility. In Section 7 we describe our prototype implementation and evaluate the performance of the proposed approach. We overview related work in Section 8. We provide concluding remarks in Section 9.

2 THE PAIRSE PROJECT: BACKGROUND

The approach presented in this paper is implemented as a part of PAIRSE project which deals with the privacy preservation issue in P2P data sharing environments, particularly in epidemiological research where the need of data sharing is apparent for making better a health environment of people. To support the decision process, epidemiological researchers should consider multiple data sources such as the patient data, his social conditions, the geographical factors, etc. The data sources are provided by DaaS services and are organized with peers. DaaS services differ from traditional Web services, in that they are stateless; i.e., they only provide information about the current state of the world but do not change that state. When such a service is executed, it accepts from a user an input data of a specified format (“typed data”) and returns back to the user some information as an output. DaaS services are modeled by RDF views.

Fig. 1 summarizes the architecture of this project. The Multi-Peer Query Processing component is in charge of answering the global user query. The latter has to be split local queries (i.e., sub-queries) and has to determine which

---

1. This research project is supported by the French National Research Agency under grant number ANR-09-SEG-008, and available at: https://picoforge.int-evry.fr/cgi-bin/twiki/view/Pairese/Web/
privacy rules \textit{Rules Set}(RS). We define a privacy rule by a topic, domain, level, and scope.

The \textit{topic} gives the privacy facet represented by the rule and may include for instance: the resource recipient, the purpose and the resource retention time. The “purpose” topic states the intent for which a resource collected by a service will be used; the "recipient" topic specifies to whom the collected resource can be revealed. The \textit{level} represents the privacy level on which the rule is applicable. The domain of a rule depends on its level. Indeed, each rule has one single level: “data” or “operation”. The \textit{domain} is a finite set that enumerates the possible values that can be taken by resources according to the rule’s topic. For instance, a subset of domain for a rule dealing with the right topic is (“no-retention”, “limited-use”). The \textit{scope} of a rule defines the granularity of the resource that is subject to privacy constraints. Two rules at most are created for each topic: one for data and another for operations.

A Privacy Rule \(R_i\) is defined by a tuple \((T_i, L_i, D_i, S_c_i)\) where:

- \(T_i\) is the topic of \(R_i\),
- \(L_i \in \{\text{“data”}, \text{“operation”}\}\) is the level of the rule,
- \(D_i\) is the domain set of \(R_i\); it enumerates the possible values that can be taken by \(T_i\) with respect to \(rs\),
- \(S_c_i\) is the scope of \(R_i\) where \(S_c_i = \{\text{“total”}, \text{“partial”}\}\) if \(L_i = \text{“operation”}\) and \(S_c_i = \{\text{“total”}\}\) if \(L_i = \text{“data”}\).

Example 1. We give two examples of rules \(R_1\) and \(R_2\) in RS, where: \(R_1 = (T_1, L_1, D_1, S_c_1)\) where \(T_1 = \text{“recipient”}\), \(D_1 = \{\text{public, research - lab, government, hospital, university}\}\) and \(L_1 = \text{“data”}\) and \(S_c_1 = \text{“total”}\). \(R_3 = (T_3, L_3, D_3, S_c_3)\) where \(T_3 = \text{“retention”}\), \(D_3 = [0, 1, \ldots, \text{Unlimited}]\) (defining retention in day), \(L_3 = \text{“data”}\) and \(S_c_3 = \text{“total”}\).

3.3 Privacy Assertion

The services will use privacy rules to define the privacy features of their resources. The application of a rule \(R_i = (T_i, L_i, D_i, S_c_i)\) on \(rs\) is a privacy assertion \(A(R_i, rs)\) where \(rs\) has \(L_i\) as a level. \(A(R_i, rs)\) states the granularity of \(rs\) that is subject to privacy. The granularity \(g\) belongs to the scope \(S_i\) of the rule. \(g\) is equal to partial if only the ID of the operation invoker is private. \(A(R_i, rs)\) also indicates \(D_i\)’s values that are attributed to \(rs\). Let us consider the rule \(R_1\) given in example 1. A privacy assertion on \(rs\) according to \(R_1\) may state that \(rs\) will be shared with government agencies and research institutions. We use the propositional formula \((p f) \land \text{“government”} \land \text{“research”}\) to specify such statement.

A Privacy Assertion \(A(R_i, rs)\) on a resource \(rs\) is defined by the couple \((pf, g)\); \(pf = v_{ip1} \land \ldots \land v_{ipq}\) according to \(R_i = (T_i, L_i, D_i, S_c_i)\), where \(v_{ip1}, \ldots, v_{ipq} \in D_i\); \(g \in S_c_i\) is the granularity of \(rs\).

3.4 Privacy Policy

A service \(S\) will define a privacy policy, \(PP_S\), that specifies the set of practices applicable to the collected resources. Defining the privacy policy \(PP_S\) of \(S\) is performed in two steps. First, the service \(S\) identifies the set (noted \(P\)) of all
privacy resources. Second, S specifies assertions for each resource rs in P_R. Deciding about the content of PR and the rules (from RS) to apply to each resource in P_R varies from a service to another. PR_R specifies the way S treats the collected resources (i.e., received through the mediator). We give below a definition of privacy policy.

The Privacy Policy PR_R of S is defined as PR_R = {A(R, rs_k), j∈[PR_R], i∈[RS], k∈[P_R], rs_k ∈ RS}.

3.5 Privacy Requirements

A service S will define a Privacy Requirements PR_R stating S’s assertions describing how S expects and requires a third-party service should use its resources. Through privacy requirements, S applies its the right to conceal their data (i.e., output).

Before creating PR_R, S first identifies the set (noted P_R) of all its privacy resources related to its output parameters and operation invocation. PR_R assertions describe the way S expects T to treat the privacy of input data, output data (e.g., experiment results returned by a service), and information about operation invocation. In addition, S may unequally value the assertions specified in PR_R. For instance, S owns SSN and zip_code data, S’s requirements about SSN may be stronger than its requirements for zip_code. Besides, S may consider an assertion more essential than another, even if both assertions are about the same resource. For that purpose, S assigns a weight w_j to each assertion A(R, rs) in PR_R; w_j is an estimate of the significance of A(R, rs). The higher the weight, the more important is the corresponding assertion. Each weight is decimal number between 0 and 1.

- \( \forall j \in [PR_R] : 0 < w_j \leq 1, \)
- \( \sum_{j=1}^{k} w_j = 1, \) where \( k = |PR_R| \).

In the real cases, S may be willing to update some of their privacy requirements. To capture this aspect, S stipulates whether an assertion A(R, rs) is mandatory or optional via a boolean attribute M_j attached to assertion A.

The Privacy Requirements PR_R of S is defined as PR_R = {A(R, rs_k), w_j, M_j, j∈[PR_R], i∈[RS], k∈[P_R], rs_k ∈ P_R, R ∈ RS, w_j is the weight of A_j, M_j = True iff A_j is mandatory}.

Example 2. Let us consider the previous rules of example 1 R_1 and R_2 and let us consider services S_11 and S_12 in Table 1, P_1 of S_11 = {SSN} and P_2 of S_12 = {SSN}. S_11 defines its PR as: PR_{S_11} = {(A_1(R_1, SSN) = hospital), (A_2(R_3, SSN) = 10)}. S_12 defines its PP as: PP_{S_12} = {(A_1(R_1, SSN) = research, lab), (A_3(R_3, SSN) = 70)}.

3.6 Privacy Annotation for WSDL-Based DaaS

In our previous work detailed in [22], we have defined a mechanism to annotate WSDL 2.0 descriptions under the interface element that describes the abstract part of the service with privacy specification of service. We choose to annotate WSDL descriptions at the three following places: interface, operation, input and output. Furthermore, we note that services are located in Peer-to Peer environment which is controlled and managed by a superpeer. A service S wanting to adhere to this environment, has to undertake to respect its PR and PP by the signing of an e-contract with the responsible peer.

4 THE PRIVACY COMPATIBILITY CHECKING

In this section, we introduce the notion of compatibility between privacy policies and requirements. Then, we define the notion of privacy subsumption and present our cost model-based privacy matching mechanism.

4.1 Privacy Subsumption

Let us consider a rule \( R_i = (T_i, L_i, D_i, S_i) \). Defining an assertion \( A(R_i, rs) = (p_f, g) \) for \( rs \) involving assigning value(s) from \( D_i \) to the propositional formula \( p_f \) of A. The values in \( D_i \) are related to each other. For instance, let us consider the domain \{public, government, federal tax, research\} for a rule dealing with topic \( T_i = “recipient” \). The value public is more general than the other values in \( D_i \). Indeed, if the recipient of \( rs \) is declared public (i.e., shared with any entity), then the recipient is also government and research. Likewise, the value government is more general than research since the research is a government agency. To capture the semantic relationship among domain values, we introduce the notion of privacy subsumption (noted \( \sqsubseteq \)). For instance, the following subsumptions can be stated: government \( \sqsubseteq \) public; research \( \sqsubseteq \) government. Note that privacy subsumption can be different from the typical subsumption of domain concepts represented with the notation \( \sqsubseteq \).

4.1.1 Privacy Subsumption

Let \( D_i = \{v_{i1}, \ldots, v_{im}\} \) be the domain of a privacy rule \( R_i \). We say that \( v_{ip} \) is subsumed by \( v_{iq} \) or \( v_{iq} \) subsumes \( v_{ip} \), (1 ≤ i ≤ m and 1 ≤ q ≤ n) noted \( v_{ip} \sqsubseteq v_{iq} \) iff \( v_{iq} \) is more general than \( v_{ip} \).

We generalize the notion of privacy subsumption to assertions. Let us consider an assertion \( A(R_i, rs) = (p_f, g) \) representing an expectation of \( S \) (resp., T) and another assertion \( A'(R_i', rs') = (p_f', g') \) modeling a practice of \( T \) (resp., S). In order for \( A \) and \( A' \) to be compatible, they must be specified on the same rule \( R_i = R_i' \), the same resource \( rs = rs' \), and at the same granularity \( g = g' \). Besides, the expectation of \( S \) (resp., T) as stated by \( p_f \) should be more general (i.e., subsumes) than the practice of \( S \) (resp., T) as given by \( p_f' \). In other words, if \( p_f \) is true, then \( p_f' \) should be true as well. For instance, if \( p_f = “government \wedge research” \) and \( p_f' = “government” \), then \( p_f \Rightarrow p_f' \) (where \( \Rightarrow \) is the symbol for implication in propositional calculus). Hence, \( A \) is more general than \( A' \) or \( A \) subsumes \( A' \) (noted \( A \sqsubseteq A' \)). Although some literals used in \( p_f \) are syntactically different from the ones used in \( p_f' \), they may be semantically related via subsumption relationships. For instance, let us assume that \( p_f = “public \wedge research” \) and \( p_f' = “federal tax” \). Since federal tax \( \sqsubseteq \) public, we can state that public \( \Rightarrow \) federal tax. In this case, we can prove that \( p_f \Rightarrow p_f' \) and hence, \( A \sqsubseteq A' \).

Then, if we consider \( A(R_i, rs) = (p_f, g) \) and \( A'(R_i', rs') = (p_f', g') \), \( A' \) is subsumed by \( A \) or \( A \) subsumes \( A' \), noted \( A \sqsubseteq A' \), if \( R_i = R_i' \), \( rs = rs' \), \( g = g' \), and \( p_f \Rightarrow p_f' \).
4.2 Privacy Compatibility Matching Algorithm

We propose an algorithm (Algorithm 1 below) called PCM (Privacy Compatibility Matching), which is previously discussed in [30], to check the privacy compatibility of PR and PP. Then, for each rs in P_out = rs’ in P_in, PCM checks the compatibility of assertions in PR5 (related to rs) with assertion in PP’ (related to rs' of S') based on the privacy subsumption described above. PCM outputs the set of incompatible assertions couple (inC). PCM matches expectations in PR5 to practices in PP’ and expectations in PP’ to practices in PR5. Two options are possible while matching PR5 and PP’. The first option is to require full matching and the second is partial matching. Indeed, the mediator may opt for the second matching type in case when some service are willing to sacrifice their privacy constraints. For that purpose, we present a cost model-based solution to enable partial matching. The cost model combines the notions of privacy matching degree and threshold. Due to the large number and heterogeneity of DaaS services, it is not always possible to find policy PP’ that fully matches a S’s requirement PR5. The privacy matching degree gives an estimate about the ratio of PR5 assertions that match PP’ assertions. We refer to M ⊂ PR5 as the set of all such PR5 assertions. The degree is obtained by adding the weights of all assertions in M: Degree (PR5, PP’) = \sum w_j for all assertions (A_j(R_i, r_{s_i}), w_j, M_j) ∈ M. The privacy matching threshold gives the minimum value allowed for a matching degree. The value of T is given by the service and gives an estimate of how much privacy the service is willing to sacrifice.

Algorithm 1: PCM

\begin{verbatim}
input : PR5={(A_j(R_i, r_{s_i}), j ≤ |PR5|, i ≤ |RS|, k ≤ |P_i|, r_{s_i} ∈ P_i, R_i ∈ RS)} (assertion of privacy requirements)
input : PP’={(A_j’(R_i, r_{s_i}’), j’ ≤ |PP’|, i ≤ |RS|, k ≤ |P_i|, r_{s_i}’ ∈ P_i, R_i ∈ RS)} (assertion of privacy policy)
output : inC (The set of incompatible assertion couple);

1) foreach r_{s_i} = r_{s_i}’ do
2) for i = 1, i ≤ |RS| do
3) for j = 1, j ≤ |PR5| do
4) for j’ = 1, j’ ≤ |PP’| do
5) if (A_j(R_i, r_{s_i}) ⊆ (A_j’(R_i, r_{s_i}’)) then
6) A_j(R_i, r_{s_i}) is compatible with A_j’(R_i, r_{s_i}’)
7) else inC ← (A_j(R_i, r_{s_i}), A_j’(R_i, r_{s_i}’))
\end{verbatim}

5 PRIVACY-AWARE COMPOSITION

The result of a composition is a set of component DaaS services which must be composed in a particular order depending on their access patterns (i.e., the ordering of their inputs and outputs parameters). In this Section, we explain our approach, which previously detailed in [30], to check the privacy compatibility within composite services.

5.1 Service Dependency in a Composition Plan

The mediator returns initially, as a result of composition, a set CP of DaaS composition plans (with CP = {CP1, CP2, . . . , CPn}), all answering the same query. The selected services, in a given CPi ∈ CP, need to be executed in a particular order depending on their inputs and outputs parameters. Note that input parameters begins with “$” and output parameters by “?”. To construct the composition plan the algorithm [2] establishes a dependency graph (noted DG) in which the nodes correspond to services and the edges correspond to dependency constraints between component services. If a service S_i needs an input x provided from an output y of service S_j then S_i must be preceded by S_j; we say that there is a dependency between S_p and S_q (or S_q depends on S_p). Fig. 2 depicts the DG of the composition plan represented related to Q. In what follows, we explain how we check the privacy compatibility of all services in DG.

5.2 Checking Privacy Within Composition

We extend the previous composition approach to deal with the privacy-preserving issue within composition. Let us consider a graph DG, if S_i depends on S_p, then S_i is showed as a consumer to some data provided by S_p and the latter is showed then as a producer from the mediator point of view. Then, the mediator considers the privacy requirements PR5 of the service producer (i.e., S_p, since PR5 specifies S_p conditions on the usage of its data) and privacy policy PP’ of the service consumer (i.e., S_q, since PP’ specifies S_q usage on the collected data) and checks the compatibility of PP’ and PR5 by using the privacy compatibility matching algorithm PCM within services order in DG. Then, a given CP is considered as privacy-preserving aware composition plan if the privacy compatibility related to all dependencies in DG are fully satisfied. In other words, if it exists at least one dependency in CP for which PR and PP of related services are not compatible, then CP is violated privacy and will be withdraw from the set CP. rs consumer with PP matching producer PR having incompatibility results in the denial of rs divulgation. The mediator can opt for a partial compatibility between PR and PP (according to the cost-model described in Section 4.2) if the concerned services with PR allow that.

Example 3. Let us consider the DG of Fig. 2 which is one of possible CP for Q. The mediator identifies firstly, from DG, service consumers, producers and resources related to each dependency step. The s parameter is an input for S_2, S_3 and S_1 while it is an output of S_1 and therefore S_2, S_3 and S_1 depend on S_1. Similarly, z is an input of S_1 and an output of S_3, therefore S_3 depends on S_1. Consequently, S_2 and S_1 are considered as consumers services, while S_1 is considered...
once as a consumer (step 1) once as a producer (in step 2 it provides output for others services). The same reasoning is observed for \( S_{3:1} \). In step 1, the mediator checks the compatibility of \( PR_{\text{input}} \) and \( PP_{S_{3:1}} \) related to \( rs = \text{"Patient Disease"} \). In step 2, the mediator checks the compatibility of \( PR_{S_{3:1}} \) and \( PP_{S_{5:1}}, \) \( PR_{S_{5:1}}, \) and \( PP_{S_{5:1}} \), where \( rs = \text{"SSN"} \). In step 3, \( rs = \text{"zip code"} \) and the compatibility of \( PR_{S_{5:1}} \) and \( PP_{S_{5:1}} \) is checked. Thus, the compatibility of \( PP_{S_{3:1}} \) and \( PP_{S_{5:1}} \) at step 2 is not hold according \( PCM \) algorithm, since hospitalresearch – lab and 1070 then Inc = \{(A1, A1), (A3, A3)\}.

### 5.3 Discussion

The compatible \( CP_{l} \) may not be entirely protected and be subject to some attacks [15] in order to disclose the identity of data which are resulted from the composition execution. We believe that a robust privacy criterion should take adversary knowledge into consideration. However, this problem is out from the scope of this paper and will be presented in a future work. The mediator has only the responsibility to response query through composition plan while assuring the compatibility between services privacy specification in \( CP_{l} \). In case where all \( CP_{l} \) of \( CP \) are incompatibility, the mediator should attempt an alternative response mechanism and avoid the empty response. In the next section, we propose a novel approach to achieve compatibility based on the negotiation taking into account the privacy.

### 6 Negotiation to Reach Compatibility

In the previous section, we showed how privacy is checked within composite services using the dependency graph and \( PCM \) algorithm. The mediator basically discards any composition plan which is subject to privacy incompatibility from the set response \( CP \). We intend (to help scientists in achieving their epidemiological tasks) to avoid such empty set response (i.e., \( CP \neq \emptyset \)) in order to improve the usefulness of the system. The main idea behind avoiding empty responses is to reach a compatible \( CP_{l} \) through a privacy-aware PP negotiation mechanism, i.e., negotiation is not achieved at the expense of privacy. In [29], we presented an early idea of privacy requirement-negotiation which is designed to offer incentives to component services in order to adapt their \( PR \).

Compared to [29], in this paper, we revise the previous idea of negotiation and provide many improvements. First, the negotiation decision is cautiously taken according to a utility-based cost function defined by a service provider. Second, the negotiation is processed with the objective to adapt the privacy policy \( PP \) of service subject to incompatibility and not its privacy requirements \( PR \). Also, we provide many additional experimental results to show the effectiveness of our proposed techniques. In the following, we detail our privacy-aware approach that aims at dynamically reconciling incompatible services’ privacy policies while always respecting the privacy requirements.

#### 6.1 Privacy-Aware Negotiation

In services composition (cf. Section 2), a mediator selects one service from several candidate services to perform a sub-part of the user query. Several approaches in literature use non-functional (QoS i.e., quality of service) properties to select services [1], [35], where the web services provide contracts that can guarantee a certain level of QoS. Contract compliance is usually assessed through a reputation mechanism. We use a similar notion to define a non-functional property called composition reputation as a criterion to select services during composition. Composition reputation (or simply, reputation) is defined as the number of times that a service \( S \) has accepted to adapt its \( PP \), divided by the number of times \( S \) received \( PP \) adaptation requests from the mediator. The more \( S \) is willing to adapt its \( PP \), the higher is its reputation

\[
\text{Reputation} (S) = \frac{N_{\text{Adapt}}(PP)}{Q_{\text{Adapt}}(PP)}
\]  

where \( N_{\text{Adapt}}(PP) \) is the number of adaption made by \( S \) on \( PP \) and \( Q_{\text{Adapt}}(PP) \) is the number adaptation requests received by \( S \) from the mediator. A service provider should generally be flexible when it specifies its \( PP \) (to attain better reputation). Moreover, a service may be willing to adapt some of its assertions in a \( PP \) while maintaining a minimum privacy level. The approach works as follows. If the \( PR_{S} \) and \( PP_{S} \) are not compatible in a given \( CP_{l} \), the related service consumer \( S_{c} \) is informed by \( PCM \) about the assertions in its \( PP_{S} \) that are incompatible. The mediator starts the negotiation process with \( S_{c} \) with the objective of achieving adaptation of \( PP_{S} \).

Fig. 3 gives an overview of the negotiation process, which is guided by the offers sent by the mediator to \( S \) and the willingness of \( S_{c} \) to negotiate its \( PP_{S} \). The Reputation-based Privacy negotiation Module (RPM) allows the mediator to decide whether a candidate \( S \) is chosen or not depending on \( \text{Reputation} (S) \). A mediator that requests a service for composition, provides feedback on the service interaction afterwards. The negotiator component handles the negotiation process by creating instances of both mediator (M—proxy) and service consumer \( S_{c} \) (C—proxy) to reach a mutually compatible solution. In what follows, we detail our negotiation approach.

#### 6.2 Negotiation Strategies Specification

In this section, we describe why, when and how a service providers and mediators define their negotiation strategies respectively.

##### 6.2.1 Why Negotiating Privacy Policies

A “good” Web service can be essentially described as a service that participates more often in compositions, that
does not disclose private data, and that does not attempt to alter data or operations. Thus, a primary reason for service providers to adopt their PPs (i.e., negotiate them) is the fact that PP should not be an obstacle (in terms of privacy incompatibility) for the services’ invocations in compositions. In other words, PP should not jeopardize the paradigm of the service use, since the more a service is utilizable, the more its reputation will grow [23]. However, this does not mean in any way that a PP be relaxed to the point where it may be compromised.

When a service provider specifies its PP, it takes into consideration (in addition to the privacy features and their impact) other features that may assist in improving its performance. Studies have demonstrated how personal data, such as information captured by the index of desktop user-trace, local analyses, etc. can be used in order to provide personalization of service functionality [31]. These personalization techniques, based on personal information, have demonstrated the potential of greatly improving the relevance of displayed service behavior. However, the sensing and storage of such information may conflict with PR of other services (see Section 5). Then, PP negotiation seems as a useful mechanism for increasing the services’ composition reputation. Obviously, the foremost challenge then is how a service provider can take the best decision between keeping its PP unchanged or negotiating it. For this, we define a utility-based cost function on privacy-efficiency trade-off, noted $C_{S_{Ne}}^{RK}$ in order to measure the gain earned by negotiating PP$_{S}$, ($U^{PP}_{REP}$), and the gain to keep PP$_{S}$ ($U^{PP}_{PR}$).

Our cost function $C_{S_{Ne}}^{RK}$ is inspired from the models proposed in [19], [17] and defined as follows:

$$C_{S_{Ne}}^{RK} = \psi(U^{PP}_{REP} - U^{PP}_{PR}).$$  \hspace{1cm} (2)

Then, a provider uses the formula (2) to evaluate the estimation of the best choice between $U^{PP}_{REP}$ and $U^{PP}_{PR}$.

### 6.2.2 How to Negotiate Privacy Policy

Guided by $C_{S_{Ne}}^{RK}$, S’s provider defines negotiation strategies beforehand when $U^{PP}_{REP}$ is greater than $U^{PP}_{PR}$. The provider also specifies an alternative assertion set PP$_{N}$ which is a subset of PP$_{S}$ (i.e., PP$_{N}$ ⊆ PP$_{S}$) related to one or several privacy rules $R_i$ for which S is willing to negotiate (i.e., for them $U^{PP}_{REP} ≥ U^{PP}_{PR}$. Each assertion in PP$_{N}$ is negotiable. Hence, PP$_{N} = \{(A_n, R_i, s_{l_i})\}, i \leq |RS|, k \leq |PP|, s_{l_i} \in P_p, R_i \in RS\}. For each $A_n$ in PP$_{N}$, S defines a negotiation strategy, noted as $S_{A_n^{Tran}}$ as one or several alternative assertions $A^p$ that alternate $A_n$. $S_{A_n^{Tran}}$ is specified as a state diagram where the initial state represents $A_n$ in PP$_{N}$ and each other state represents an alternate assertion $A^p$. Each transition between states represents an accepted offer which is described as an incentive $I^p$. Thus, $S_{A_n^{Tran}} = \{I^p(A^p), 1 \leq p \leq |S_{A_n^{Tran}}|\}$.

Fig. 4 illustrates the $S_{A_1^{Tran}}$ negotiation strategy ($S_{A_1^{Tran}}$) defined for assertion $A_1$ (with respect to (2)). According to $S_{A_1^{Tran}}$, $S_{A_1}$ accepts to negotiate its initial assertion $A_1$ (of $S_{A_1^{Tran}}$). Then, if $S_{A_1}$ receives the incentive $I^1$, it will change $A_1$ = “Research – lab” as recipient to $A_1$ = “Federal – tax”. Otherwise, it adapts $A_1$ to $A_3$ = “Hospital” if it will receive the incentive $I^3$.

### 6.2.3 Mediator Negotiation Strategy

The mediator is central to the collaborative negotiation strategy. S’s provider informs mediator about its $C_{S_{Ne}}^{RK}$. The mediator then defines its negotiation strategies. Since mediator is considered as a trusted entity, it consults the value of $C_{S_{Ne}}^{RK}$ of S only if S appears as an incompatible service for a composition plan. Thus, according to $C_{S_{Ne}}^{RK}$ of S, the mediator identifies a sub-set of privacy rules, noted as RS$_{N}$, for which it is willing to negotiate with S. Then, for all the rules $i \in RS_{N}$, the mediator defines a negotiation strategy which is guided by the set of incentives. Each negotiation strategy can be described as a state machine where each state represents an incentive $I^i$ and each transition between states represents a not accepted response to $I^i$ that may be returned from S. We assume that the mediator knows the initial reputation value of S (noted $V_{REP}(S)$). $V_{REP}(S)$ measures the trustworthiness of S based on end-user feedbacks. $V_{REP}(S)$ corresponds to the average of collected ratings and can be quantitatively measured. Based on RPM, the mediator initially defines, regarding RS$_{N}$, a finite set of offers $Ofr = \{I^1, ..., I^n\}$, (with $n = |Ofr|$). The set of offers is ordered and $I^1 = V_{REP}(S)$ with $I^j < ... < I^n$. Each incentive $I^j$ (where $1 < j < n$) is defined as the increase proportion value $\in [1\% \text{, } 100\%]$ of the original service reputation value. The more the incentive is important, the more service reputation value will be increased. The ranking of incentives to be sent to S is illustrated according to a negotiation strategy. The proportion value of incentive, noted as percent $Rep$, that mediator increases between the states of a negotiation strategy, is calculated as

$$\%Rep = \frac{F_i(S) + c * Reputation(S)}{1 + c}$$  \hspace{1cm} (3)

where

$$F_i(S) = (V_{REP}(S), V_{AVG}(S))$$  \hspace{1cm} (4)

$F_i(S)$ represents $F_i(S)$ average which is a vector of feedback values for S computed from the last $\theta$ queries in which S was invoked. $V_{REP}(S)$ represents the initial reputation value of S and $V_{AVG}(S)$ is the probability that S was available for the corresponding query. The $c$ parameter of formula (3) is a weighting factor assigned to the composition reputation (formula (1)). The mediator assigns more importance to the reputation than feedback values (of formula (4)), thus $c > 1$. The mediator negotiation strategy is described as: $M_{St}^{Rep} = \{I^1 \succ I^2, 1 \leq q \leq |M_{St}^{Rep}|, I^p, I^q \in Ofr, p < q, R^i \in RS_{N}\}$.

Fig. 5 illustrates a mediator negotiation strategy regarding $R_1$. The mediator will update its negotiation strategy...
only when: 1) $S$ will be invoked in a new CP and $S$ will not be compatible, and, 2) $S$ will send a new update of its $C_S^{N-C-K}$. Other services in the composition that are willing to negotiate are not able to discover the negotiation strategies of the mediator.

### 6.3 Negotiation Protocol

We propose a dynamic protocol called ReP (Algorithm 2), handled by the negotiator module. This protocol aims at automatically reconciling the mediator’s and consumer’s negotiation strategies related to consumer assertions in InC. In this regard, the negotiation protocol incorporates two state machine diagrams using the reconciliation algorithm, and finds the first alternative assertion from $S_{Tran}^{1}$ that is compatible with $A_w$. The algorithm ReP checks if an incentive $I^1$, from $M_{Stat}^{1}$, is accepted by $S_{Tran}^{1}$ and then checks the compatibility of the related alternative assertion $A^q$ (instead of $A_w$) from $S_{Tran}^{1}$, (where the couple $(A_w, A^q) \in \text{InC}$). Otherwise, if $A^q$, related to the acceptance of $I^1$, is not compatible with $A_w$, the algorithm ReP will check the next incentive from $M_{Stat}^{1}$; looks if it is accepted by $S_{Tran}^{1}$ and the previous reasoning is observed. Thus, ReP is applied to all assertion couples (related to consumer services) of InC under the condition that there exist negotiation strategies specified for each assertion (of the corresponding privacy policy) of InC. The algorithm ReP returns $\text{Rec}$ which contains the best alternative assertions that will be compatible. A successful negotiation concludes with a mutually agreed and signed policy, called privacy e-agreement contract (between concerned service and mediator).

**Example 4.** Let us consider the negotiation strategies of Figs. 4 and 5 and the assertion couple $(A_1, A_1)$ of $\text{InC} = \{(A_1, A_1), (A_3, A_3)\}$. These strategies are specified regarding $R_1$. Then, according to the algorithm ReP, the first $I^1$, is accepted but $A^1$ related to $I^1$ is not compatible with $A_1 = \{\text{"hospital"}\}$. ReP retrieves the second offer from $M_{Stat}^{1}$, i.e., $I^2$. This latter is not accepted by $S_{Tran}^{A_3}$, then ReP retrieves the third $I^3$ which is accepted by $S_{Tran}^{A_3}$. The related $A^3$ of $I^3 = \{\text{"hospital"}\}$ and it is compatible with $A_1$.

### 7 Prototype and Evaluation

The goal of our experiments is twofold: first, we study the performance of the proposed algorithms and protocols via extensive experiments. Second, we validate the applicability of our proposal on real-life scenarios.

We first describe the prototype architecture in Section 7.1. We detail the experiments setup in Section 7.2. In Sections 7.3 and 7.4, we study the performance evaluation of the proposed algorithms (privacy compatibility checking, PCM, and negotiation, ReP respectively). Then, in Section 7.5, we report our experiment results with three real scenarios from the healthcare domain to show the impact of PCM and ReP algorithms on service composition time processing, including server-side time consumption and client-side total response time.

#### 7.1 Prototype Architecture

Our prototype allows querying and composing DaaS according to the architecture depicted in Fig. 6, which is organized into four layers. The first layer contains a set MySQL databases that store medical data. The second layer includes a set of proprietary applications developed in Java; each application accesses databases from the first layer. These proprietary applications are exported as DaaS services. These services constitute the third layer, and their description files (i.e., WSDLs) are annotated with RDF views and published via registries (we use Openchord DHT to this end). The upper layer includes a Graphical User Interface (GUI) and a Web Service management system (WSMS). The GUI component is composed of two basic interfaces: Requester-Interface and Administrator-Interface. Users access the system via Requester-Interface of the GUI to submit queries to the composition system. Administrator accesses the system to develop and manage Web services through the Privacy Composition Checking and Privacy Adaptation components, which implement our PCM algorithm and negotiation process respectively (see Fig. 3). The Requester interface of our prototype is available at http://soc.univ-lyon1.fr:8080/queryRewriter/index.html. It can be downloaded and executed with the Java Web Start technology, and it relies on a locally deployed DHT based on OpenChord to store the descriptions of DaaS services.

#### 7.2 Experiments Set-Up

We realized two classes of experiments. The first class evaluates the compatibility and negotiation approaches
(cf. Sections 7.3 and 7.4 respectively). We used the deployment kit bundled with GWT (Google Web Toolkit) and the Apache server Tomcat to develop and deploy the prototype. We run these experiments on a laptop with 2.53 GHz Intel Core 2 duo processor with 4 Go of RAM, and under the Mac OS X 10.6.8 operating system. The performance has been measured in terms of CPU time (in milliseconds). We measured the average CPU time for 30 iterations of our PCM and ReP algorithms. We noticed that after 10 iterations the average value becomes stable.

The second class is related to real life scenarios (cf. Section 7.5). We implemented the DaaS services involved in the scenarios on a virtual machine hosted on the Lyon 1 university campus. The virtual machine has been granted the following hardware characteristics: 64 bit Intel single core CPU at 2.66 Ghz with 1 Go RAM. The network our experiment have been tested on is a 1000BASE-T switched full-duplex network, deployed with Cat-5 twisted pair cables. We connected to the network with RJ45 cables and Gigabit PCI Ethernet cards.

### 7.3 Privacy-Compatibility Evaluation

In the PAIRSE prototype, we developed more than 100 real Web services. The developed services include services providing medical information about patients, their hospital visits, diagnosed diseases, lab tests, prescribed medications, etc. In the following, we evaluate the efficiency and scalability of our compatibility algorithm.

For each service deployed in our architecture, we randomly generated PR and PP files regarding its manipulated resources (i.e., inputs and outputs). Assertions in PR and PP were generated randomly and stored in XML files. All services were deployed over an Apache Tomcat 6 server on the Internet. We implemented our PCM algorithm in Java and run the composition system with and without checking compatibility. To evaluate the impact of PCM on the composition processing, we performed two sets of experiments.

#### 7.3.1 Efficiency and Scalability

In the first set of experiments, we mainly focused on the compatibility checking phase with the perspective to evaluate the effectiveness and speed of the PCM. The computational complexity of PCM algorithm is of the order $O(n^2)$. Indeed, the total number of assertions that must be checked among $PR_S$ (containing $n$ assertions) and $PP_S$ (containing $m$ assertions) with respect to one dependency step in $CP$ (i.e., between $S$ and $S'$) is equal to $n \times m$. Hence, our PCM has a polynomial complexity. In order to empirically verify this assumption, we conducted a set of experiments to analyze the scalability of PCM as the sizes of PP and PR increase. Fig. 7a shows the performance of the PCM as the PP and PR file sizes (noted as $|PR|$ and $|PP|$ respectively) increase. The experiment is processed on two files PP and PR. Then, when $|PP| = 18$ and $|PR| = 18$ assertions the time is around 60 ms. For $|PP| = 36$ and $|PR| = 36$ assertions, the processing time is 240 ms. Then, when the size $|PR|$ and $|PR|$ is doubled, the execution time increases 4-fold. Thus, for $|PP| = 72$ and $|PR| = 72$, the processing time is close to 960 ms.


7.3.2 Impacts of Dimensionality

In the second set of experiments, we evaluated the impact of 
CP size (i.e., \(|CP|\): the number of services in 
CP) on the 
PCM processing time. For that purpose, we generated 
synthetic 
CPs and varied the number of services in each 
generated 
CP. In the first experiment, each service in any 
generated 
CP had \(|PP| = 10\) and \(|PR| = 10\) assertions. In the 
second experiment, each service in any 
CP had \(|PP| = 20\) and \(|PR| = 20\) assertions. Fig. 7b shows 
the performance of 
PCM as the composition size increases for 
both experiments. We can argue that the time of 
PCM is 
linear with respect to the size of 
CP. However, comparing 
the two experiments in Fig. 7b, the processing time of 
PCM is 
polynomial with respect to the number of assertions 
of each service in 
CP. We take as example two different \(|CP|\) 
(\(|CP| = 30\) services and \(|CP| = 60\) services) and we compare 
the proportion of increase in 
PCM time processing. For 
\(|CP| = 30\) services with \(|PP| = 10\), \(|PR| = 10\) of each service 
in that 
CP, the processing time is near to 740 ms. Similarly, for 
\(|CP| = 60\) services with \(|PP| = 10\) and \(|PR| = 10\) of each 
service in that 
CP, the processing time is near to 867 ms. Fig. 7 
allows us to confirm that in general when the size of 
CP is 
doubled the execution time is increased by a factor of 
less than 1.7. For the same \(|CP| = 30\) services, with each service 
having \(|PP| = 20\), \(|PR| = 20\) of each service in that 
CP, the 
processing time is near to 2900 ms. For \(|CP| = 60\) services 
having \(|PP| = 20\), \(|PR| = 20\) for service in that 
CP, the 
PCM processing time attains 3430 ms. Overall, the impact of 
\(|CP|\) on the 
PCM processing time is less important than that of 
\(|PP|\) and \(|PR|\).

7.4 Negotiation Performance

In the following we evaluate the performance of our 
negotiation approach. We first describe the case of 
incompatibility considered by the negotiation approach, 
before presenting and discussing the most significant 
results obtained from our experiments. The negotiation 
proposal deals with the case of privacy incompatibilities 
between services within a composition plan. Two services 
S and S' within a 
CP (where S' depends on S) are 
incompatible in terms of privacy regarding a dependent 
resource rs if PR does not subsume PP for that 
rs. In this case, the negotiation can be performed to reach a 
compatible 
CP. Note that other reasons for privacy incompatibility 
can exist: 1) If rs \(\notin PP\) and rs \(\notin PR\) then, PP and PR are 
not compatible, 2) If rs \(\in PP\) and rs \(\notin PR\) then, PP and PR are 
not compatible, and 3) S' does not have PP, S' is 
considered as incompatible regarding any other service.

The three previous cases of incompatibility are not 
considered by the negotiation approach.

We implemented our ReP algorithm in Java. For the sake 
of performance study, for each developed service we 
randomly generated negotiation strategies. Each strategy 
STran is attached to the corresponding assertion, which 
is related to Retention topic and is defined on DT = [1, . . . , 100]. 
On the other side, we randomly generated a set of negotiation 
strategies MStat of the mediator where R = Retention topic. 
Each negotiation strategy of the mediator is defined to one 
corresponding service. All the negotiation strategies 
are stored in XML files. We analyzed the time performance of 
ReP as the size of the set MStat increases (i.e., \(|MStat|\): 
the number of offers). Fig. 8a shows the time to compute the 
adapted value of ReP. The results obtained show that even 
for a large number of offers (e.g., 100), the negotiation time 
remains negligible (344 milliseconds for 100 offers). Fig. 8b 
shows the performance of 5 negotiation processes related to 
5 services (into the same incompatible 
CP) at the same level of 
dependency graph. Each service strategy is defined on 
Retention-assertion topic and contains 10 possible states (i.e., 
|TA| = 10) where the set of offer of the mediator 
negotiation strategy varied from 10 to 100. The execution 
times are close; which confirms the capability of the 
approach to carry out several negotiation processes in 
parallel.

7.5 Validation via Scenarios

We evaluated the impact of our solution with three 
scenarios, noted as Sce1, Sce3, and Sce3, respectively. 
They reflect typical use cases of the application domain. 
The following three scenarios have been proposed by one 
of our partners, the Cardiology Hospital of Lyon, in the 
PAIRSE project.

- Sce1. The first scenario Sce1 involves 5 services 
(PatientByIDService, CurrentTreatmentByPatientIDService, 
MedicalHistoryByPatientIDService, MedicationByTreatmentIDService and 
DrugClassByMedicationService) and 3 service dependencies. This scenario returns the 
different risks associated with the patient’s current 
and previous treatments, along with a description of 
the patient’s profile. It is mainly useful for doctors to 
monitor their patients’ history and helps for treat- 
ment prescription.

- Sce2. The second scenario Sce2 involves 3 services 
(CurrentTreatmentByPatientIDService, MedicationByTreatmentIDService and 
DrugClassByMedicationService)
and 2 service dependencies. It gives the risks associated to a patient’s current treatment. It is useful for nurses to help understand the patients’ problems for daily care.

- **Sce3.** The third scenario Sce3 involves 2 services (PatientByIDService) and 1 dependency. It returns the description of a patient. It is useful for administrative staff to manage patients’ information (i.e., mail for invoice).

The services involved in these scenarios are: PatientByIDService takes as input a patient ID to return the patient’s description. CurrentTreatmentByPatientIDService takes a patient ID to return the patient’s current treatment. MedicalHistoryByPatientIDService takes a patient ID to return the patient’s previous treatments. MedicationByTreatmentIDService takes a treatment ID to return the medication involved in this treatment. DrugclassByMedicationService takes a medication ID to return the drug class of this medication (indicates the different risks to be associated to the medication). We performed two sets of evaluations, and measured the results obtained with and without negotiation. Each set of run has been executed 30 times, at which point the results seem to converge.

Tables 2 and 3 show the end-to-end latency timings as seen by the requester of the three previous scenarios. Each line shows (timings in columns 3 to 6 are in milliseconds in both tables). Column (1) indicates if the composition plans are compatible without negotiation (C), with negotiation (C (neg)) or not compatible (N – C), Column (2) indicates the number of service combinations in each CP generated by the system to answer the query of the scenario, Column (3) indicates the mean time the system takes to answer the query, Column (4) indicates the minimum execution time to answer the query, Column (5) indicates the maximum time to answer the query, and Column (6) gives the standard deviation of the timings obtained. Timing in Table 3 are computed on the same three previous scenarios with other services while PR/PP of these services are different from services used in Table 2.

The results obtained with the scenarios show that the overhead of negotiation to reach compatible CP is low (up to 30 percent overhead for Sce1 and Sce3, which explains the strongest impact), which confirms our results obtained in Section 7.4. Compared to the experiments performed in Sections 7.3 and 7.4, the scenarios analyzed in this section show a higher variation between the minimal and maximal latency. We interpret such a result as being due to the significance of the network latency. Most of the time is spent retrieving WSDL, PR/PP and negotiation strategies files over the network, thus making the execution times of our algorithms much smaller than the global response time.

| TABLE 2 | Client-Side Timings of the Different Scenarios |
|---|---|---|---|---|---|---|
| Sce1 | N – C | 216 | 1034.1 | 202.0 | 3998.0 | 663.3 |
| Sce1 | C (neg) | 216 | 1302.6 | 400.0 | 6280.0 | 826.4 |
| Sce2 | N – C | 72 | 675.8 | 296.0 | 2207.0 | 349.5 |
| Sce2 | C (neg) | 72 | 776.3 | 396.0 | 2413.0 | 371.6 |
| Sce3 | N – C | 64 | 149.0 | 36.0 | 1915.0 | 247.0 |
| Sce3 | C (neg) | 64 | 217.5 | 127.0 | 2172.0 | 374.3 |

| TABLE 3 | Client-Side Timings of the Different Scenarios |
|---|---|---|---|---|---|---|
| Sce1 | C | 216 | 1281.8 | 205.0 | 9372.0 | 866.8 |
| Sce2 | C | 72 | 401.0 | 178.0 | 1538.0 | 176.9 |
| Sce3 | C | 64 | 137.3 | 42.0 | 1466.0 | 187.2 |

Such a result indicates our solution has a low overhead and is applicable to the scenarios developed in the context of the PAIRSE project.

### 7.6 Limitations

We argue that a compatible composition plan (regardless of the way to obtain it) is not entirely protected. Several types of attack [15] can be carried out against composition execution $T_{CP}$ (where $T_{CP}$ being the table of the compatible CP execution) in order to re-identify published data. We need to evaluate how much information can be inferred with respect to the attacker’s knowledge. The solution we deem the most appropriate is to efficiently model the attacker’s knowledge through several dimensions with the perspective to calculating the probability for an adversary to re-identify the data contained in $T_{CP}$. Our goal will be to prevent the adversary from predicting whether a target individual $t$ (contained in $T_{CP}$) has a target sensitive value $s$.

### 8 RELATED WORK

We review the closely related areas below and discuss how our work leverages and advances the current state-of-the-art techniques.

#### 8.1 Privacy Model Specification

A typical example of modeling privacy is the Platform for Privacy Preferences (P3P) [34]. However, the major focus of P3P is to enable only Web sites to convey their privacy policies. In [32] privacy only takes into account a limited set of data fields and rights. Data providers specify how to use the service (mandatory and optional data for querying the service), while individuals specify the type of access for each part of their personal data contained in the service: free, limited, or not given using a DAML-S ontology. In [27], Ran propose a discovery model that takes into account functional and QoS-related requirements, and in which QoS claims of services are checked with external components that act as certifiers. The authors refer to the privacy concern with the term confidentiality, and some questions are raised about how the service makes sure that the data are accessed and modified only by authorized personals. Some policy languages, such as XACML [25], ExPDT [8] are proposed and deployed over a variety of enforcement architectures.

These languages are on the one hand syntactically expressive enough to represent complex policy rules, and offer on the other hand a formal semantics for operators to reason about policies, e.g., their conjunction and recently difference. Unfortunately, they do not provide solution when an incompatibility occurs. In our work, privacy resource is specified and may be related to client, Data and Service providers levels, and not only to the provided data.
8.2 Privacy-Aware Composition

The works in services composition are closely inspired from workflow and Data mashups composition. In [5] a framework for enforcing data privacy in workflows is described. In [6], the use of private data is reasoned for workflows. Privacy-preserving mechanism for data mash-up is represented in [20]. It aims at integrating private data from different data providers in secure manner. The authors in [13] discuss the integration and verification of privacy policies in SOA-based workflows. The previous approaches, related to data mashup and workflows, focus on using algorithms (such as k-anonymity) for preserving privacy of data in a given table, while in our work we go further and propose a model that also takes into account usage restrictions and client requirements. The work [7] proposes using third parties as database service providers without the need for expensive cryptographic operations. However the proposed schemes do not allow queries to execute over the data of multiple providers and do not take into account the privacy issue regarding service provider and data consumer, which is the main focus of our work. In [9], privacy leakage in multi-party environment has been investigated. The approach takes a game-theoretic approach to analysis some of privacy assumption in the presence of colluding parties. It consists of a light-weight method to let each participant estimate the percentage of colluders in the environment. However, the secure multi-party based-methods involve a high computational cost in distributed system. One appealing approach is described in [4] and aims at preserving privacy of private data mashup with the social networks. The issue this approach resolves, is to dynamically integrate data from different sources for the joint data analysis in the presence of privacy concerns.

In contrast to the existing approaches, our privacy model described in this paper goes beyond “traditional” data-oriented privacy approaches. Input/output data as well as operation invocation may reveal sensitive information about services and hence, should be subject to privacy constraints.

8.3 Privacy and Negotiation

The proposal of [12] is based on privacy policy lattice which is created for mining privacy preference-service item correlations. Using this lattice, privacy policies can be visualized and privacy negotiation rules can then be generated. The Privacy Advocate approach [14] consists of three main units: the privacy policy evaluation, the signature and the entities preferences unit. The negotiation focuses on data recipients and purpose only. An extension of P3P is proposed in [11]. It aims at adjusting a pervasive P3P-based negotiation mechanism for a privacy control. It implements a multi-agent negotiation mechanism on top of a pervasive P3P system. The approach proposed in [26] aims at accomplishing privacy-aware access control by adding negotiation protocol and encrypting data under the classified level.

Previous work are suffering from two major shortcomings: The first one is the “take-it-or-leave-it” principle, i.e., a service can only accept or refuse the other service’s proposal as a whole. The second is the “one-size-fits-all” principle: once the service producer has designed its privacy policy, it will be proposed to all interested services no matter what their requirements are. Our privacy model goes beyond previous privacy approaches and aims at ensuring privacy compatibility of involved services in the composition without any additional overload. Moreover, it reconciles the incompatibility of privacy concerns using a negotiation protocol.

9 CONCLUSION AND FUTURE WORK

In this paper, we proposed a dynamic privacy model for Web services. The model deals with privacy at the data and operation levels. We also proposed a negotiation approach to tackle the incompatibilities between privacy policies and requirements. Although privacy cannot be carelessly negotiated as typical data, it is still possible to negotiate a part of privacy policy for specific purposes. In any case, privacy policies always reflect the usage of private data as specified or agreed upon by service providers. As a future work, we aim at designing techniques for protecting the composition results from privacy attacks before the final result is returned by the mediator.

ACKNOWLEDGMENT

The authors would like to thank: P. De Vettor for his contribution to the development of the experiments, and J. Fayn for her help in the realization of the scenarios.

REFERENCES


Salah-Eddine Tbahriti received the PhD degree in computer science from the Claude Bernard Lyon 1 University, Lyon, France, in 2012. He is a Member of LIRIS CNRS Laboratory. His research interests include privacy preservation in service composition, privacy preserving, and network security. Dr. Tbahriti has published several papers on Web service privacy in international journals and conferences such as IEEE-Systems Journals, ICWS, and IEEE-WI.

Chirine Ghedira is currently a Full Professor of computer science at Jean Moulin Lyon 3 University, Lyon, France. She was a Member of the LIRIS Laboratory till 2011. Her research interests include distributed information systems, Web services, and context-aware computing. Ms. Ghedira has served on numerous conference program committees and has organized different scientific events (e.g., Notere 2008, CWS-05, CINC 2005, etc.).

Brahim Medjahed received the PhD degree in computer science from Virginia Tech, Blacksburg, in May 2004. He is an Associate Professor of computer science at the University of Michigan-Dearborn, Dearborn. His research interests include service-oriented computing, distributed computing, and semantic Web. Dr. Medjahed has served on numerous conference program committees. He is the author of more than 60 publications.

Michael Mrissa received the PhD degree from the Claude Bernard Lyon 1 University, Lyon, France, in 2007. He is an Associate Professor of computer science at Claude Bernard Lyon 1 University and a Member of the LIRIS CNRS Laboratory. His main research interests are related to Web services, data and information management, and semantic Web. His publication list includes international journals and conferences such as ACM TOIT and ER.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.