Privacy Aware Service Selection of Composite Web Services Invited Paper

Anna Cinzia Squicciarini*, Barbara Carminati#, Sushama Karumanchi*
*College of Information Sciences and Technology
The Pennsylvania State University, University Park, PA 16802.
E-mail: {asquicciarini,sik5273}@ist.psu.edu
#Department of Theoretical and Applied Science
University of Insubria, Varese, Italy.
E-mail: barbara.carminati@uninsubria.it

Invited Paper

***Abstract*—Web service selection involves finding services from a possibly huge database of similar services. Hence, the challenges involved in finding a suitable service include large time consumption, and difficulty of finding a perfect match according to the user specified search keywords. In addition, significant privacy and security concerns arise, as the information involved with service selection and provisioning may be sensitive for both providers and users.**

In this paper, we propose a comprehensive framework to uniformly protect users and service providers at the time of service selection. We define a solution that (a) supports private service selection, such that both criteria and service attributes are kept private during the matching and (b) includes an approach to protect service provisioning rules from unwanted disclosure, both from the user and the service provider's perspective. We implemented our solution and integrated with the current Web service standard technologies. We conducted an extensive experimental evaluation, using datasets of actual WSDL documents. Our experimental evaluation and complexity analysis demonstrate that our algorithms are both effective and efficient.

***Index Terms*—Web services, Privacy, Selection and Composition**

I. Introduction

The growing success of Web services as the preferred standards-based approach to implement distributed and heterogeneous systems has resulted in a large number of providers offering services of varying degree of sophistication and complexity, ranging from weather forecast to travel management. On one hand, the availability of a wide array of services has created a competitive and flexible market that meets the needs of different type of users (e.g. individuals, organizations, law enforcement etc.). On the other hand, this large availability of heterogenous services makes the service selection process a non-trivial task, making Web service selection and provisioning play a crucial role in Web service life-cycle. Here, several application-dependent requirements might constrain the selection of the best service.

Existing literature offers some approaches toward an efficient and effective selection of Web services, where interesting solutions are proposed in the context of Web service recommendation [1], [2], [8], [12], [18]. However, several important problems have not been addressed yet. First, while all existing efforts confirm the relevance and need of a constrained service selection approach, wherein users can specify their preferences and needs, none of them takes into account the privacy of both users and service providers. In particular, existing works fail to recognize that even optimized strategies for service selection involve the exchange of large amount of potentially sensitive data, causing potentially serious privacy leaks. Further exacerbating the risk of privacy leaks is the need of creating composite Web services to meet complex users' needs.

In addition to considering the privacy requirements associated with the disclosure of user's personal data needed to locate Web services according to his/her *search criteria*, we also take into account the fact that service providers might have to verify a user's profile information against some *provisioning rules*, prior to service invocation. These rules could be imposed by law, or by internal business rules. For example, in the context of e-commerce transactions, laws protecting against selling alcohol to minors might have to be applied. Or in the case of a rental car service, the provider might prefer to lend luxury cars only to clients with a given income, or it might choose not to lend them to young drivers. All these conditions have to be retrieved and matched against the user's profile prior to service provisioning. Similar to the case of search criteria, these rules might have to be verified against private/sensitive user's attributes (i.e., age and income), therefore giving rise to privacy problems.

Henceforth, not only end users may have privacy and confidentiality needs, but also Web service providers may need to maintain provisioning rules, possibly confidential. As such, *privacy and information leakage* associated with service provisioning are also important. This problem is again compounded by the fact that multiple Web service providers may be involved, and a dynamic composition process is to be carried out on the fly. In the presence of multiple providers that coordinate together to provide a composite Web service, privacy issues are amplified, due to the larger number of service providers involved (each of which may have individual

COLLABORATECOM 2013, October 20-23, Austin, United States
Copyright © 2013 ICST
DOI 10.4108/icst.collaboratecom.2013.254264

provisioning rules).

Motivated by the above considerations, in this work we present a comprehensive framework for secure selection and provisioning of Web services. Our goal to help users and services obtain and deliver composite and single services in a private fashion. Our framework uniformly protects users' and service providers' privacy needs, at the time of service selection. We define a solution that performs matching of the user search criteria against the Web services attributes in a private fashion such that both criteria and service attributes are kept private during the matching.

To provide the advanced functionalities offered by our architecture, we implement a prototype of our solution integrated with the current Web service standard technologies. Moreover, we conduct experimental evaluation using real WSDL documents. As demonstrated by our empirical and theoretical evaluation, our algorithms are efficient with respect to both time and computation. A preliminary version of this framework has been proposed in [15], where we only focused on the private matching of service provider provisioning rules and user profile, and private user search. In this work, we develop a private service selection framework with new methods, to ensure that only services that have capabilities that highly suit the requirements of the user are selected, in a private fashion. Our extended solution also includes algorithms to deal with complex private rules of Web Service providers, as well as a description of a standard-compliant architecture of the system. Our experimental evaluation is comprehensive, and includes a large set of experiments on all the proposed algorithms.

The paper is organized as follows. In Section II, we provide an overview of the relevant literature in the area. In Section III, we describe our proposed framework, and elaborate on its most important protocols. We present the language used for representation of rules in Section IV. In Section V, we present the private service selection and provisioning model. In Section VI, we discuss our prototype and report some experimental evaluation. We conclude in Section VII.

II. Related Works

Work related to the proposed system is associated with the following main topics: access control, privacy for Web services, and Web service composition. We review relevant efforts in these areas in what follows.

Access control in Web services focuses on authentication and authorization of the users or the service providers requesting a Web service, or processing a Web service in case of Web service composition. To date, a number of works providing solutions for access control in Web services have been proposed [7], [16], [11], [20], [14]. In what follows, we discuss few of the most representative approaches. Few works focus on attribute based access control in Web services [7], [14], [20] where the access control to resources is achieved based on the attributes of the subjects, resources and the environment under which the access takes place, and it is therefore well suited for Web services. For example, in [20], the authors posit that ABAC is a natural convergence of the following existing access control models: (1) discretionary access control (DAC) whose implementations include identity based access control, role based access control, (2) mandatory access control (MAC). They claim that ABAC is more fine-grained and the policy representation is richer and more expressive than the other access control models. Also, authors in [16] propose to extend WS-BPEL, so as to allow specification of user tasks and their enforcement of access control of those tasks.

Parallel to work on access control, researchers have investigated how to address the client's privacy concerns [11], [10], [17]. In [11], a framework focusing on clients' privacy is proposed. The key component of their framework is the mobile agent which ensures accurate data privacy policy of the service provider to be implemented at the client. Similarly, authors in [10] describe an approach to prioritize the Web services for a client during selection depending on the privacy preferences specified by the client. Work to date has taken into consideration the privacy of the client's information during the service provisioning phase and later, but do not consider the client's information privacy during the service selection phase. Users' search criteria information is as vulnerable to secondary usage as other data of the client stored at the service provider's side. During the selection of Web services, the client needs to provide its search criteria to the UDDI registry entity or a search engine. To business critical clients this search information is very confidential. In our work, we provide a solution to protect the search criteria of the client from disclosure to the service provider.

Web service composition is a related important and evolving paradigm, as it provides better services to the clients. Web service composition involves composition of different Web services on the fly, upon request from a client. Composition brings with it a new set of privacy problems to the service providers. During service composition, business related and sensitive information of the service providers needs to be exchanged among themselves in order to provide the composite service. On a first step towards this problem direction, authors in [13], consider the privacy problem that arises among the service providers during Web service composition. They analyze the problem of information flow during Web service composition, whereby service providers might derive some sensitive information during the composition process. Ustun and colleagues [19] provide an approach to manage the information flow between composed services in a decentralized manner using peer-to-peer processes. As in [13], they make use of the information flow policies and build algorithms to achieve the goal. Both these solutions fail to consider the privacy of these information flow policies from the other services participating in the Web service composition. In [21], the authors focus on controlling access to data resources from a client by making the composite Web service combine the access control policies of different Web services and construct its own set of access control policies for the client requesting the composite service. However, the access control policies of each Web service are disclosed to the other services. While similar to the extent that we also deal with composing policies from multiple providers, we focus on privacy of access control policies of each service provider during Web service composition. In [4], the authors present a framework for Web service composition that is decentralized supporting and enforcing some security requirements like availability, confidentiality, authenticity and integrity of the execution order of the Web services. Carminati and colleagues [5] mainly provide an idea about how to compose Web services taking into consideration their security constraints and capabilities, and posit that the service composition should be made only after checking the compatibility of different Web service provider's security constraints and capabilities. In our work, we propose not only to consider the compatibility of security constraints and capabilities among the service providers, but we also consider various business policies and their confidentiality needs.

III. Overall Architecture

We recall that the goal of the proposed framework is to assist users to privately search for services able to carry on the required business, as well as to privately enforce possible provisioning rules of the selected service providers. For this purpose, the proposed architecture has a main component called the Private Negotiator (PN, in short). The Private Negotiator enables private search over services, and also provides a private profile compatibility verification method that takes place among service providers, and between the service provider and the user. As such, the user search for the services is carried out in two distinct steps. The first step involves the user providing the planner (discussed later) non-private information on the service he is looking for (i.e., *public search*), whereas the second step involves the user providing the Private Negotiator, the private information, such as location or implementation requirements, for the *private* search and provisioning rule enforcement. We discuss these two steps in more detail in what follows.

Step 1: As the first step, the user enters information regarding the required service, that is, the description, the business type, its input and output (Figure 1, step 1). For example, the user provides "Travel Service" as description of the service, "airline, hotel, and payment services" as the types of services, "input:date" and "output:cost" as the input and output of the service. A planner module [6] performs a search on the UDDI registry for the service requested by the user (step 2). After the search, the planner module prepares a list of composite Web service combinations that suit the user request and sends this list to the PN (step 3).

Step 2: The Private Negotiator (PN) enables the user to privately search for services from the received list, and the service provider and the user to privately evaluate their provisioning rules and profiles against each other. It also enables the same among the service providers in case of a composite service. In order to support private search as well as private enforcement of provisioning rules, in [15] we proposed the use of the Private Matching Protocol (PM). The Private Matching Protocol has been introduced by Malkhi in [9], to allow two parties to learn in a private fashion the set of common attributes. Each of the two parties holds a database, and wishes to jointly calculate the intersection of their inputs, without leaking any additional information. The exchange is asymmetric, so that only the party initiating the PM algorithm learns the result of the matching.

The PM protocol is activated when the user submits to PN a private search request to be evaluated on the list of services returned by the planner. The user typically provides private information to PN as the search criteria. For example, the user could search for "services processing Visa credit cards" and "services that take input only from US citizens".

Upon receiving such type of request, the PN coordinates a PM protocol between the user and the service provider on the list of services returned by the planner, so as to evaluate private search criteria conditions against the profiles of the services in a private fashion. As further elaborated in Section V-A, executing the two-party protocol allows users to identify a suitable service combination without having to reveal in clear any information related to their private search criteria. After evaluating the private search criteria, a second list of service compositions is provided to the user, and the user selects one composition among these.
The service combination selected by the user is then further processed by the PN so as to evaluate whether the associated provisioning rules of the services of the composition can be satisfied, according to the user profile. In the case of a service selection offered by a single provider, the provisioning rules are enforced using a similar approach as the one used for private search criteria. That is, a two-party private intersection protocol between the service provider provisioning rules and user profile data is carried out (rather than service descriptions, as in the case of user search criteria). However, in case the user requires a composite Web service, the private enforcement of provisioning rules could be a complex task since the number of rules to be satisfied could be high, and so would be the number of required interactions of the two-party private intersection protocol for the end user. To address this problem, as described in [15], our approach minimizes the number of provisioning rules to be evaluated by the user by executing a multi-party private intersection protocol on the service providers' rules. To handle both the private matching protocols (step 6 of Figure 1), PN is equipped with two modules: the *two-party coordinator* (PM-2P) that evaluates private search criteria and provisioning rules for simple service selection (see Section V-A) and the *multi-party coordinator* (PM-MP) that evaluates provisioning rules in case of multi-party service (see Section V-B). To facilitate and coordinate the usage of PN, clients and servers should communicate with each other through the Private Negotiator at the client side, and the Private Negotiator at the server side, as discussed in Section V.

Example 1: As an example, consider a user requesting a travel service which includes airline reservation (SP_1), hotel reservation (SP_2) and payment services (SP_3). As such, the travel service is a composite service with three types of

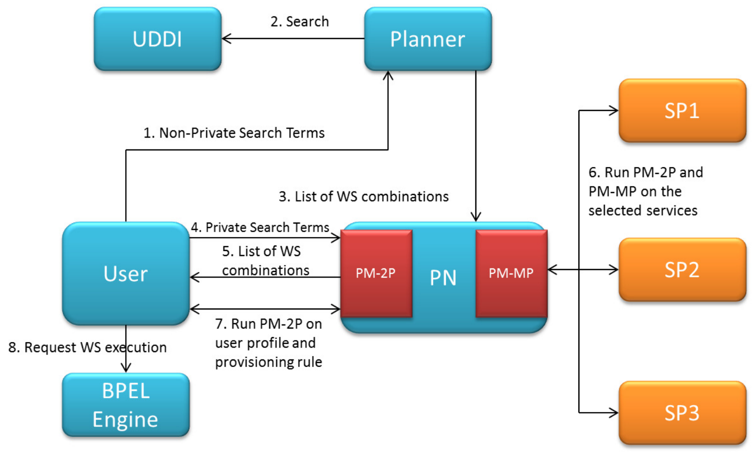
UDDI
2. Search
Planner
1. Non-Private Search Terms
3. List of WS combinations
SP1
6. Run PM-2P and PM-MP on the selected services
4. Private Search Terms
5. List of WS combinations
User
PM-2P
PN
PM-MP
SP2
7. Run PM-2P on user profile and provisioning rule
8. Request WS execution
BPEL Engine
SP3

Fig. 1: Overall architecture of the selection and provisioning framework

services. The user provides the description of the type of service he or she desires, that is, a travel service, along with the dates of travel, that should be between May 20th of 2013 and June 15th of 2013. Additionally, he indicates that he is looking for a flight which cost is below 500$ and he wants the transaction to occur over a secure and encrypted channel. Ideally, the user would like to be informed about the possible Web services options available to him. Once a suitable service or combination of services is provided, the user is likely to have to disclose service-specific information to properly customize the service selection. In this case, this information includes the user's citizenship, form of payment, travel needs etc. Given the sensitive nature of this information, the user is concerned about his privacy and wishes his/her data to be handled in a confidential fashion. Hence, he allows disclosing contact information, demographic data and financial information, but he wants it to be retained from the travel company for at most 30 days. He is not willing to release his record to third party companies.

The providers offering this complex service need to gather the users' profile, while on the one hand maximizing the chance of successful provisioning and on the other hand, minimizing the amount of information disclosed with the other service providers offering the Web service.

IV. User Search Criteria and Service Provisioning Rules

Both the user private/public search criteria and service provisioning rules can be represented as boolean expression of atomic conditions, which are matched against Web service descriptions and user's profiles' attributes, respectively. In the following we first formally introduce the atomic condition, criteria and provisioning rules.

A. Criteria and Rule Conditions

We assume each Web service has a set of *capabilities*. These capabilities describe the functional/ non-functional properties of a Web service, such as pricing, estimated time of execution, adopted algorithm for data encryption, retention time of personal data, etc. Furthermore, we assume the presence of taxonomies according to which these capabilities can be grouped based on the common domain. We also assume that user profile consists of a set of *attributes* containing user's personal information such as age, city of birth, and so forth. Under these assumptions, user search criteria and provisioning rules can be defined as boolean expressions combining *atomic conditions* on profile attributes and Web service capabilities, respectively. Atomic conditions are defined by users for their private/public search criteria, and by service providers to verify their provisioning rules against user profiles. We define atomic conditions as follows.

Definition 1 (Atomic condition): Let C be the set of capabilities, T be the set of taxonomies defined on C, and A be the set of user profile attributes. An atomic condition *cond* can be in four forms: (1) $\texttt{Att}$ $\texttt{OP}$ $value_a$, $\texttt{Att} \in A$; (2) $\texttt{CAP}$ $\texttt{OP}$ $value_c$; where $\texttt{OP} \in \{=, \in, >, \geq, <, \leq, \subset, \subseteq, \supset, \supseteq\}$. $value_a$, and $value_c$ are the service provider and user preferred values for `Att` and `CAP` respectively.

Moreover, according to the requirements introduced in Section I, a user can specify a prioritization on search criteria. As such, a user search criteria can be defined as follows.

Definition 2 (User Search Criteria): A user private/public search criteria uc is represented as a pair $(exp, priority)$ where exp is a boolean expression of atomic conditions (cfr. Definition 1), whereas $priority \in [0, 1]$ represents the user-specified priority level, denoted as $pr(uc)$, associated with the user criteria.

Example 2: In the travel reservation scenario, two possible user criteria are: 'uc=$\texttt{FlightPrice}<_p500$ AND $\texttt{CryptoAlgo}=_{sup}\{\text{TDES,AES}\}$', where, for clarity, the priority levels are omitted.

Definition 3 (Service Provisioning Rule): A service providing rule ru is a boolean expression of atomic conditions only in the form (1) (cfr. Definition 1).

To simplify the computation of private matching protocols, hereafter we represent the service provisioning rule as a Disjunctive Normal Form (DNF) consisting of multiple clauses, where each clause consists of a conjunction of atomic conditions.

Example 3: Let us assume three providers are available to provide the travel service of Example 1, dealing with airline reservation, hotel reservation and payment services, respectively. One of the potential airline providers sells flights at a competitive price only to US citizens and allows up to three baggages for free. The offer is valid also for French citizens, in which case 4 bags are included in the price. The hotel provider offers rooms for customers with a minimum stay of one day who can pay through VISA credit card and have a credit score higher than 1000. Alternatively, other forms of payment are accepted, but the minimum credit score threshold is higher. These provisioning policies described are translated as follows: The airline reservation provider will have: $((Citizenship = UnitedStates)\ AND\ (Baggage = 3))$ OR $((Citizenship = France)\ AND\ (Baggage = 4))$.
The hotel reservation provider maintains the following rule: $((Citizenship = Any)\ AND\ (MinimumStay =$1day$))$, whereas, the payment service requirements are abstracted by the rule: $((Method = VISA)\ AND\ (CreditScore > 1000))$ OR $((Method = Any)\ AND\ (CreditScore > 3000))$.

V. Private Service Selection

We now discuss the role of the private negotiator and its main tasks. The private negotiator enables the process of private service selection by providing private search and private provision rule compatibility verification among the service providers, and between the service providers and the user. PN consists of two main components: two-party coordinator and multi-party coordinator, which coordinate the private search, private provision rule compatibility verification and final rule formation. The private negotiator is also installed at the client side called the PN_CLIENT plug-in, and at the service provider side called the PN_SP plug-in. The PN_CLIENT is a browser plug-in which enables the user to specify the search terms and the priority, and helps in coordinating the two party protocol. The PN_SP plug-in helps the service providers in enforcing the two-party and multi-party protocols, and hence acts as a proxy between the service and PN. Both PN_CLIENT and PN_SP are equipped with encryption functionalities. In our discussion of the two-party and of the multi-party coordinator, for the sake of clarity, we present the algorithms for the matching of individual conditions included in provisioning rules or in search conditions. Next, in section V-C, we discuss how the individual conditions are extracted to facilitate matching and reduce the computation overhead.

A. Two-party coordinator

In this section, we show how the two-party coordinator can aid the enforcement of the private search criteria, and, in case of single services, of service provisioning rules. Let us start by introducing the process for a single generic constraint evaluation, irrespective of what entity is applying the conditions to evaluate (i.e., user for search criteria or service provider for provisioning rule), and over which/whose profiles the conditions have to be matched (i.e. user/provider). In general, users or providers' conditions are defined by constraints evaluated against assertions by means of a two-party private intersection protocol. We recall that according to this protocol, given a condition $Cond$ and its range of satisfiability RS_{Cond}, the constraint owner has to (1) generate a polynomial of degree n, where n is the cardinality of RS_{Cond}; and (2) compute the homomorphic encryption of its coefficients obtaining the private range of satisfiability (i.e., PRS_{Cond}). The private range of satisfiability together with the attribute name over which the condition is posed (i.e., $Cond.name_A$) are passed to the Two-party coordinator. The Two-party coordinator asks the profile owner for the private encryption of $Cond.name_A$ value. More precisely, the profile owner retrieves the value, say $value_{A_i}$, and sends back the encryption $Enc(rP(v_a) + v_a)$. The two-party coordinator sends this encrypted value to the rule owner, that is able to decrypt and obtain $value_{A_i}$, if $value_{A_i} \in RS_{Cond}$, a random number otherwise. Notice that as a result of this process, only the entity enforcing the conditions is able to see the outcome of private matching protocol, while the profile owner does not learn anything about the outcome. These steps are executed by both PSE_CLIENT and PSE_SP module. More precisely, in case of search criteria PSE_CLIENT and PSE_SP play the constraint and profile owner, respectively, and vice-versa in case of provisioning rule. In any case, the constraint owner is the initiator of the two-party process, and is the one who learns the outcome of the process, while the profile owner is the one that just provides the input to the two-party process. The same property applies in case of multi-party processes. These features are key to increase the secrecy of information among the participants, and will allow only the necessary participants to know the outcome of the processes.

After selecting a clause from each DNF of a SP, two party and multi-party private set intersection protocols (Sections V-A and V-B) are performed over the attributes in the clauses. That is, each service provider uses its final clause during the provisioning rule compatibility check coordinated by the two-party and multi-party coordinators.

B. Multi-Party Coordinator

We now focus on the case where a user selects a composite service. Here, provisioning rules of multiple service providers have to be combined together for actual service provisioning. Our approach is to combine rules applied by the providers in a private fashion, before presenting them to the user. This strategy has two main benefits: first, it minimizes the number of provisioning rules that the user is eventually asked to comply to, and, second, it minimizes the amount of information disclosed by service providers (in terms of provisioning rules) to clients and competing providers. Each provider enforces one provisioning rule expressed as a DNF, from which a clause is selected as explained in Section VI-A. We also assume that the workflow is modeled as a sequence of activities. The following example is used throughout the section to clarify the main steps.

Example 4: For the travel service from Example 1, consisting of the WF of three activities for which there are three services from distinct service providers, namely $SP1, \ldots, SP3$.Assume that each service provider has a clause whose conditions are defined over a set of attributes or capabilities, denoted as SP_j_ATT, where SP_1_ATT: {Age, Cost, Bags, Weight} SP_2_ATT: {Age, Cost, Salary, Hours}, SP_3_ATT: {Age, Cost, Salary, Bags}.
Let us consider in particular clauses specified over Age and let $RS_{Cond\_Age\_SP_j}$ be the range of satisfiability of condition imposed by rule at SP_j side. The following conditions hold. $RS_{Cond\_Age\_SP1} = \{1, 8, 9\}$; $RS_{Cond\_Age\_SP2} = \{1, 9, 7, 6\}$; $RS_{Cond\_Age\_SP3} = \{5, 1, 9\}$.

The overall process is as follows. In brief, the algorithm has two main phases: the first one allows individual service providers to identify the common set of attributes used by other service providers in their clauses. The second part instead consists of determining whether or not the conditions where the common attributes appear have overlapping ranges of satisfiability. If no overlap among the common attributes exists, i.e. if a common satisfiability range cannot be determined, the conditions of the clauses are incompatible with each other, and hence the service providers' rules are not compatible.

In order to complete these tasks, the service providers run on-cascade two-party set intersection protocols with other providers in the WF. In addition, to find common satisfiability ranges, multi party set intersection private protocols are carried out.

The multi-party set intersection aims at comparing the range of satisfiability of conditions defined on the attribute. The providers essentially compute the intersection among $RS_{Cond_{X\_SPi}}$ and $RS_{Cond_{X\_SPj}}$ with $j \in [1, n]$, where X is the common attribute and n is the number of services in the WF. The actual multi-party set intersection protocol builds on the Beaver-Micali-Rogaway cryptographic protocol, that operates on a Boolean circuit representation of the computed function (see [3] for details). As in the case of two-party intersection, the outcome of this protocol, that is, the acceptable satisfiability range, is known only to the service provider who initiated the multi-party private set intersection process.

Example 5: To compute the acceptable satisfiability range on conditions with domain 'Age', SP_1 initiates a multi-party private set intersection process with SP_2, SP_3, and SP_4. In this process, each service provider SP_j, $j \in \{1, \ldots, 4\}$ participates by passing encrypted version of its $RS_{Cond_{Age\_SP_j}}$.Thus, the outcome of the multi-party private set intersection process initiated by SP_1 on domain 'Age' is the set {1,9}. Similarly, multi-party private set intersection process is conducted on every common attribute with other service providers.

At the end of each and every computation, the multi-party coordinator obtains a message with the resulting outcome. When all the individual conditions are evaluated for a given SP, the SP also indicates whether or not the provisioning rule of the service provider is satisfiable. Further, it receives attribute names and corresponding conditions not in common with others. Upon receiving input from all the providers, the multi-party coordinator generates a unique provisioning rule composed by the set of conditions (i.e., a unique set of range of satisfiability) identified for each common attribute whose range of satisfiability is computed as intersection, plus those conditions defined over non common attributes. The resulting provisioning rule is then evaluated against user's profile mediated by the two-party coordinator. If the set intersection protocols do not produce a final rule over the selected clauses (that is, the clauses have not complied with each other meaning there are no common values for at least one attribute of two clauses), we re-execute the clause selection algorithm to select one clause from each SP trying with a different clause than last time.

C. Clause selection from a non-atomic rule

We now discuss how to relax the assumption made in the previous section, that is, each service provider has a single provisioning rule defined by a single atomic clause. In reality, service providers (and users as well) are likely to maintain non-atomic rules wherein conditions are combined using disjunctions and conjunctions operators. In this case, each actor (e.g. user or provider) at each round needs to identify the atomic condition to match first. It is highly likely that if clauses are randomly selected for matching by each service provider, the compatibility match may fail, even if other compatible clauses exist. For example, if a given provider SP' has a rule ($Age > 40$ AND $Salary > 4000$) OR ($Cost < 100$), and SP" has a rule ($Age > 10$ OR $Cost > 300$), by matching $Cost < 100$ and $Cost > 300$ the providers will find an empty intersection of acceptable values, leading to a halt or additional compatibility checks despite the existence of two other compatible clauses ($Age > 40$ AND $Salary > 4000$) and $Age > 10$, respectively). To avoid these pitfalls, the proposed approach attempts to maximize the chance of matching compatible clauses .

We provide a step-by-step informal discussion of the overall algorithm through an example. We consider the scenario from Example 1, where SP_1, SP_2, and SP_3 are the airline reservation, hotel reservation, and payment services respectively. We consider that the services have the following provisioning rules.
SP_1: ($Age < 60$ AND $Cost < 15$) OR ($Cost < 15$ AND $Weight > 35$ AND $Salary < 5000$);
SP_2: ($Age < 30$ AND $Cost < 25$) OR ($Cost < 20$ AND $Weight < 65$) OR ($Days > 5$ AND $CreditScore > 60$);
SP_3: ($Age < 55$ AND $Salary > 2000$ AND $Hours > 40$) OR ($Cost > 50$ AND $Bags < 5$)

1) The first service provider, SP_1, selects a clause from its

provisioning rule which is the shortest among all other clauses of its rule, that is, a clause with lesser number of atomic conditions (e.g. $Age < 60$ AND $Cost < 15$).

2) SP_1 adds the attributes to an array, $\mathcal{V}$. Array $\mathcal{V}$ is used to add the attributes of the clauses from all the service providers which are likely to be in the final provisioning rule of the composite service. That is, $\mathcal{V}$ consists of $\{Age, Cost\}$, properly indexed.
3) Next, SP_1 performs two-party private intersection with the remaining providers, one by one, over attributes in $\mathcal{V}$ and the attributes of all clauses of SP_i. For example, the two-party private intersection takes place between $\mathcal{V}$: $\{Age, Cost\}$ and {Age,Cost,Weight,Days,Credit Score} of SP_2. SP_i, as a result of the computation, only sees the result of the private intersection but not the contents of $\mathcal{V}$ in clear. SP_i selects a clause from its provisioning rule that does not contain common attributes between SP_1 and itself, and adds the attributes to $\mathcal{V}$. For example, SP_2 chooses the clause ($Days > 5$ AND $CreditScore > 60$) and $\mathcal{V}$ has {Age,Cost,Days,Credit Score}.
4) If all the clauses of SP_i have different attributes from those in $\mathcal{V}$, SP_i selects a clause randomly, and adds the attributes of the selected clause to array $\mathcal{V}$ if not in $\mathcal{V}$ already
5) If all the clauses of SP_i have at least one common attribute with those in $\mathcal{V}$, SP_i selects a clause with the same operators as those attributes in the array $\mathcal{V}$, if known (the operators may be given by SP_1). There is a higher chance of compatibility of conditions if the conditions have same operators for the common attributes, than if the conditions have dissimilar operators. In our example, SP_3 requests SP_1 for the operators of Age and Cost which are $<$ and $<$ respectively. Hence, SP_3 chooses the clause ($Age < 55$ AND $Salary > 2000$ AND $Hours > 40$) as the operator of 'Age' is the same, and it adds these attributes to array $\mathcal{V}$: {Age,Cost,Days,Credit Score,Salary,Hours}. SP_i selects a clause with higher satisfaction level of operators, that is, the clause with the higher number of operators similar to those in Age. Further, if two or more clauses have the same satisfaction level of operators, then the SP_i selects a clause with common attributes having wider range of values. This way, attributes with wider range of values have higher chance to be compatible with any condition than narrow range of values. For example, suppose that the attribute Age of SP_3 has the operator $>$, different from that in array $\mathcal{V}$, SP_3 chooses clause ($Cost > 50$ AND $Bags < 5$) as $Cost > 50$ has a range of 50-infinity which is greater than 55-infinity, the range of $Age > 55$.
6) If there are no clauses with same operators, for the common attributes, as those in $\mathcal{V}$, SP_i randomly picks a clause with an overall wider range of all of its attributes.
7) In any case, the attributes of the selected clause are added by each SP_i to the array $\mathcal{V}$, and the algorithm is repeated for the remaining SPs (that is, executed from step 3).

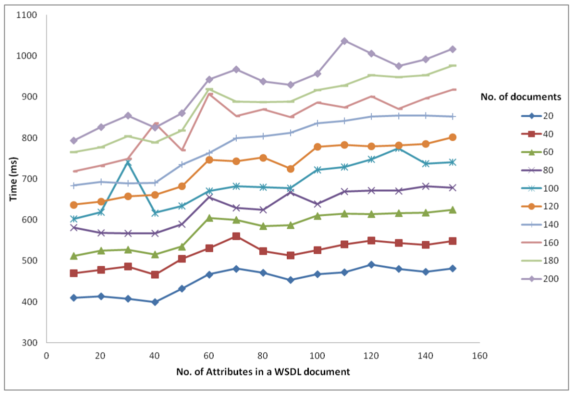

Fig. 2: WSDL parser results

VI. Experimental Results

Tests and implementation have been conducted on a Intel core 2 quad CPU, Q6700 @ 2.66GHz, 4 CPU cores, 3 GB RAM, Red Hat Enterprise Linux 5.6 operating system.
As depicted in Figure 1, the proposed framework's prototype consists of the following components: a client, set of Service Providers, a UDDI registry entity hosted by a UDDI provider, a planner module and the core component of our architecture, the Private Service Selection. The main algorithms of PN are written in a high-level definition language called Secure Function Definition Language (SFDL) adapted from [9]. The intersection function, written in SFDL, is given as input to the two-party secure function evaluation core algorithm or to the multi-party secure function evaluation algorithm [3].

In what follows, we report our results concerning the algorithms implemented by the PN.

A. *Private Service Negotiation*

WSDL Parser. We analyze the overhead incurred by running private selection of Web service, in case of simple and composite Web services. As part of the analysis of private search, we also extensively tested the overhead added by the parser, to estimate whether it significantly affects the search time for a client.

Precisely, for this experiment, we performed several rounds of tests for different set of actual WSDL documents by varying the number of attributes/search keywords in a WSDL document and tracked the time taken to parse these documents. At each round, we submitted a certain number of documents consisting of the same number of attributes -operations of the WSDL, that are the function names of the interface of a Web service- as input to the parser and recorded the time taken to parse these documents. In the same round, we considered the same number of documents but with a different number of attributes and tracked the time taken to parse these documents. In the consecutive rounds, we gradually increased the number of documents with the same test procedure, from 20 to 200, for each round as described. In Figure 2, we present the test results for this experiment. The time to parse a set of documents

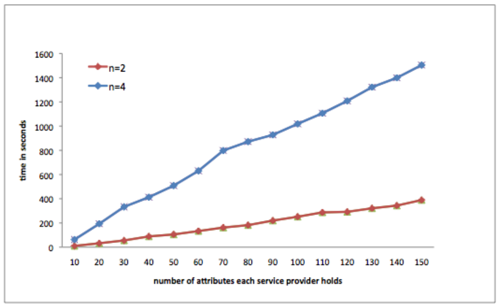

Fig. 3: Two-party set intersection, with two and four parties

slightly increases as the number of attributes in the documents increases and is always below 1000 ms (for 200 documents).

Private Selection. As far as private selection goes, we recall that enforcing search criteria and provision rules while selecting a simple service requires the execution of two-party private intersection protocol. The overall results for running two party set intersection protocols are reported in Figure 3. As anticipated, the overhead is linear with respect to the size of the set intersection input, i.e. the number of attributes considered.

To estimate the overhead added by private selection in case of composite Web services, we completed two experiments on provisioning rule composition. There are two main steps to be completed, (1) to identify the common attributes and (2) check their satisfiability ranges.

First, we calculated the time required to identify the common attribute sets. In Figure 3, the actual results for the case of a workflow with 4 providers is also considered. In this experiment, we let the service providers execute the two-party protocols in cascade. If the protocols are executed in parallel, the overall execution time is of the same order of the time for single two-party process, otherwise it increases by a factor equal to the number of providers participating in the process. Next, we considered the overhead added by computing private intersection among range of satisfiability of conditions held by multiple providers on common attribute domains. Our results show that with increasing number of service providers participating in a multi-party process, the run time of the multi-party algorithm quadratically increases. Further experiments, not reported for lack of space, also confirmed that, as the total number of common attributes among all the participating service providers increases (keeping the number of service providers constant), the run time of the multi-party algorithm grows linearly. The execution time of the multi-party processes can be substantially reduced if the parties execute the multi-party intersection protocols in parallel, rather than sequentially.

VII. Conclusion

Web service selection plays a crucial role in Web service life-cycle. Several application-dependent requirements might constrain the selection of the best service. Although there is a large body of work in service selection and composition, several issues related to privacy and security are still unsolved. Privacy, in particular, is becoming a key requirement in the selection process, from both service providers and client ends. Toward addressing this need, in this paper we presented a comprehensive framework for private selection of Web services.

We will extend the current solution along two dimensions. First, we will develop a framework that suggests security settings, by analyzing the candidate services, to the user which he or she will take into account in his or her security policies. Second, we will conduct a formal security analysis to estimate the amount of information possibly inferred by service providers and users during the protocols.

References

[1] E. Al-Masri and Q.H. Mahmoud. Qos-based discovery and ranking of web services. In *Proceedings of 16th International Conference on Computer Communications and Networks*, pages 529 –534, aug. 2007.

[2] Y. Badr, A. Abraham, F. Biennier, and C. Grosan. Enhancing web service selection by user preferences of non-functional features. In *4th International Conference on Next Generation Web Services Practices*, pages 60 –65, oct. 2008.

[3] Assaf Ben-David, Noam Nisan, and Benny Pinkas. Fairplaymp: a system for secure multi-party computation. In *Proc. of the ACM Conference on Computer and Communications Security*, pages 257–266, 2008.

[4] J. Biskup, B. Carminati, E. Ferrari, F. Muller, and S. Wortmann. Towards secure execution orders for compositeweb services. In *IEEE International Conference on Web Services, ICWS 2007*, pages 489 –496, july 2007.

[5] B. Carminati, E. Ferrari, and P.C. K. Hung. Web service composition: A security perspective. *International Workshop on Challenges in Web Information Retrieval and Integration*, 0:248–253, 2005.

[6] Daniela Barreiro Claro, Patrick Albers, and Jin-Kao Hao. A framework for automatic composition of rfq web services. *Services, IEEE Congress on*, 0:221–228, 2007.

[7] Shen Hai-Bo. A semantic- and attribute-based framework for web services access control. In *Intelligent Systems and Applications (ISA), 2010 2nd International Workshop on*, pages 1 –4, may 2010.

[8] S. Kalepu, S. Krishnaswamy, and S.W. Loke. Verity: a qos metric for selecting web services and providers. In *Fourth International Conference on Web Information Systems Engineering Workshops*, pages 131 – 139, dec. 2003.

[9] Dahlia Malkhi, Noam Nisan, Benny Pinkas, and Yaron Sella. Fairplay - secure two-party computation system. In *Proc. of the USENIX Security Symposium*, pages 287–302, 2004.

[10] Maryam Zarreh Mona Tavakolan and Mohammad Abdollahi Azgomi. Web service discovery based on privacy preferences. *International Journal of Web Services Practices*, 4(1):28–35, 2009.

[11] Abdelmounaam Rezgui, Mourad Ouzzani, Athman Bouguettaya, and Brahim Medjahed. Preserving privacy in Web services. In *Proc. of the 4th international workshop on Web information and data management*, WIDM '02, pages 56–62, New York, NY, USA, 2002.

[12] Mohamed Sellami, Samir Tata, Zakaria Maamar, and Bruno Defude. A recommender system for web services discovery in a distributed registry environment. In *Proceedings of the 2009 Fourth International Conference on Internet and Web Applications and Services*, pages 418–423. IEEE Computer Society, 2009.

[13] Wei She, I-Ling Yen, Bhavani M. Thuraisingham, and Elisa Bertino. Policy-driven service composition with information flow control. In *IEEE International Conference on Web Services, ICWS 2010*, pages 50–57, 2010.

[14] Haibo Shen. A semantic-aware attribute-based access control model for web services. In *Proceedings of the 9th International Conference on Algorithms and Architectures for Parallel Processing*, ICA3PP '09, pages 693–703, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] Anna Cinzia Squicciarini, Barbara Carminati, and Sushama Karumanchi. A privacy-preserving approach for web service selection and provisioning. In *IEEE International Conference on Web Service, ICWS 2011*, pages 33–40, 2011.

[16] J. Thomas, F. Paci, E. Bertino, and P. Eugster. User tasks and access control over Web Services. In *IEEE International Conference on Web Services, ICWS 2007*, pages 60 –69, july 2007.

[17] Arif Tumer, Asuman Dogac, and I. Hakki Toroslu. A semantic based privacy framework for web services. In *In Proc. of E-Services and Semantic Web Workshop*, 2003.

[18] Le Hung Vu, Manfred Hauswirth, and Karl Aberer. Qos-based service selection and ranking with trust and reputation management. In *Proceedings of the Cooperative Information System Conference*, pages 446–483, 2005.

[19] Ustun Yildiz and Claude Godart. Information flow control with decentralized service compositions. *IEEE International Conference on Web Services, ICWS 2007*, 0:9–17, 2007.

[20] E. Yuan and J. Tong. Attributed based access control (abac) for web services. In *IEEE International Conference on Web Services, CWS 2005*, pages 2 vol. (xxxiii+856), july 2005.

[21] Junqiang Zhu, Yu Zhou, and Weiqin Tong. Access control on the composition of web services. *International Conference on Next Generation Web Services Practices*, 0:89–93, 2006.