2012 26th International Conference on Advanced Information Networking and Applications Workshops

How to Preserve Privacy in Services Interaction

Salah-Eddine Tbahriti [1] Brahim Medjahed [2] Zaki Malik [3] Chirine Ghedira [4] Michael Mrissa [1]

[1] Université de Claude Bernard Lyon 1
LIRIS UMR 5205
Lyon, France
{firstname.lastname}@liris.cnrs.fr

[2] University of Michigan-Dearborn
Department of Computer and Information Science
Dearborn, USA
brahim@umd.umich.edu

[3] Wayne State University
Department of Computer Science
Detroit, USA
zaki@wayne.edu

[4] Université de Jean Moulin Lyon 3
IAE Laboratory
Lyon, France
chirine.ghedira-guegan@univ-lyon3.fr

Abstract — **In this paper, we present a formal model for preserving privacy in Web services. We define a Web service-aware privacy model that deals with the privacy of input data, output data, and operation usage. We introduce a matching protocol that caters for partial and total privacy compatibility. We propose also a negotiation model to reconcile clients' requirements with providers' policies in case of incompatibility.**

Keywords: Privacy, Web services, Negotiation.

I. Introduction

While initial Web service standards and technologies have been beneficial in the deployment of service-based systems, the issue of privacy has been recognized as one of the main reasons that prevent users from using the Internet for accessing on-line services [1][7]. Despite important regulatory and technical efforts aimed at preserving privacy, privacy leakage incidents on the Web continue to make the headlines [13]. Two factors exacerbate the problem of privacy in service-oriented environments. First, Web services collect and store a large amount of information about users. Second, Web services share this information with other Web services. Besides, the emergence of analysis tools makes it easier to analyze and synthesize huge volumes of information, hence increasing the risk of privacy violation [5].

By privacy we mean the right of an entity to determine on the first hand which information is considered as private and, on the second hand, why, for whom, and for how long it will release that information. We identify two types of entities in a service-to-service interaction: clients invoking a Web service (e.g., users, Web services, applications) and providers (i.e., Web service being invoked). In order to deal with the network heterogeneity, we consider that all services follow the same annotation and based on the same description. The, clients submit input data to invoke providers' operations; providers return output data to clients as results. Therefore, three categories of information are perceived as private by clients and/or providers: *input*, *output*, and *operation invocation*. On the provider side, providers may impose privacy constraints on their returned (i.e., output) data. On the client side, any input submitted by clients to providers may be subject to privacy requirements. Clients may also impose privacy constraints on outputs, although providers generate such data. Finally, clients may view their operation invocations (independently of input/output data) as sensitive, for example, a patient invoking the operation set_doctor_appointment() of a hospital's cardiology Web service. Third parties (e.g., life insurance companies) may conclude that the patient is suffering from heart conditions, if they know about this invocation. To prevent such privacy leakage, the patient may declare the operation usage as private.

Each client/provider specifies how it handles private information (i.e., inputs, outputs, and operation usage), and how it expects the other entity to treat that information. A provider WS specifies a *privacy policy* PP^{WS} that details the set of privacy usage applicable to all clients. For each provider WS, client C defines a *privacy requirements* $PR^{C/WS}$ stating C's perceptions about WS inputs, outputs, and operation usage. In reality, C may unequally value the importance of its privacy requirements in the same $PR^{C/WS}$. In addition, C may demand a full compatibility between PP^{WS} and $PR^{C/WS}$ while another client may be satisfied with partial compatibility to a certain threshold specified by the client. In the case of incompatibility between PP^{WS} and $PR^{C/WS}$, two options are possible. First, inform C and WS that their interaction cannot take place. Second, initiate a negotiation process between C and WS to reach consensus between both entities. While the former solution is easier to implement, the latter is more flexible and allows for dynamic and self-adapting privacy requirements and/or policies.

In this paper, we propose a formal model for privacy in Web service interaction. The paper's contribution focuses on the following issues:

- *Privacy Model* - We propose a Web service-aware privacy model. This model is mainly based on our previous approach proposed in [1] and takes into account features specific to Web services such as the privacy of input/output data and operation usage. Privacy policies and requirements are specified according to privacy rules that can be dynamically added, deleted, and modified by system administrators.

978-0-7695-4652-0/12 $26.00 © 2012 IEEE
DOI 10.1109/WAINA.2012.83

IEEE computer society

- *Privacy Matching* - We define a matching protocol which checks the compatibility of providers' policies and clients' requirements. The protocol is based on the notion of privacy subsumption and cost model. A matching threshold is set up by clients to cater for partial and total privacy compatibility.
- *Negotiation Model* - We introduce a negotiation model based on incentives to reconcile privacy requirements and policies in case of incompatibility. Clients and providers specify their negotiation strategies via state diagrams.

The rest of this paper is organized as follows. Section II defines the privacy model. Section III describes the privacy matching protocol. Section IV is devoted to the negotiation model. Section V summarizes related work. Section VI describes our prototype implementation. Concluding remarks are provided in Section VII.

II. Privacy Model

In this Section, we provide the reader with some foundational concepts of our privacy model. We do not further elaborate on the details and we refer the reader to [1] for a full description of our privacy model. Our privacy model is based on the definition of : *Privacy level*, *Privacy Rule*, *Privacy Assertions*, *Privacy Policy* and *Privacy Requirements*.

A. Privacy Level

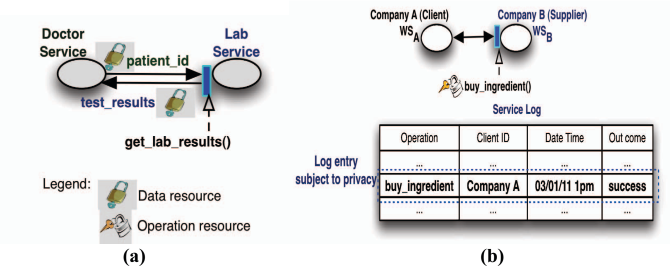
The goal of our privacy model is to protect private information. We refer to such information as *privacy resources* (simply *resources*). Each service has the ability to identify which concerned information is considered as private. We define two privacy levels: *data* and *operation*. The *data level* deals with the privacy of data shared between clients and providers (Figure 1.a). Data resources refer to the input and output parameters of a service operation (e.g., defined in WSDL). For instance, let us consider an operation that returns the lab test results performed by a patient at a certain date. The patient_id (input) and test_results (output) may be viewed as private; they are hence defined as data resources. The *operation level* copes with the privacy of operation usage/invocation. Figure 1.b depicts a business-to-business interaction between two services WS_A and WS_B representing companies A and B, respectively. Assume that WS_A invokes WS_B's buy_ingredient() operation. Company A may consider such invocation as a business trade secret: if the invocation of this operation is disclosed to A's competitors, then A may suffer insurmountable losses. Therefore, A perceives the operation buy_ingredient() as a privacy resource.

Definition 1 – Let *rs* be a *privacy resource* of a Web service *WS*. The *privacy level* L of *rs* is defined as follows: (i) L = "data" if *rs* is an input/output of a WS operation; (ii) L = "operation" if *rs* is a WS operation. ◊

Operation invocations may be perceived as private independently on whether their input/output parameters are confidential or not.

B. Privacy Rule

The sensitivity of a resource may be defined according to several dimensions called *privacy rules*. We call the set of privacy rules *Rules Set* ($\mathcal{RS}$) which is described and stored in the system administrators. Rules may be added, modified, and deleted at anytime.



Operation	Client ID	Date Time	Out come
...	...	...	...
buy_ingredient	Company A	03/01/11 1pm	success
...	...	...	...

(a) (b)

Figure 1. Privacy ressources

Definition 2 – A *privacy rule* R_i, is defined by a tuple (T_i, L_i, D_i, S_i) where:

- T_i is the topic of R_i.
- $L_i \in$ {"data", "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.
- S_i is the scope of R_i where S_i = {"total", "partial"} if L_i = "operation" and S_i = {"total"} if L_i = "data". ◊

The *scope* of a rule defines the granularity of the resource that is subject to privacy constraints. Clients and providers assign one of the values "total" or "partial" to the scope of their operation resources. If an operation resource is assigned a "total" scope for a given rule, then the whole entry of that operation in the service log is private. In the case of data rules, we consider data resources as atomic. Hence, the only scope value allowed in this situation is {"total"}. "Partial" scope may also be considered for complex data resources (e.g., array, structure). In this case, only part of an input/output parameter is private. For instance, let us consider two privacy rules R_1 and R_2 such as

- R_1=(T_1=Recipient, L_1=Data, D_1={"public", "government", "federal tax"}, S_1={"total"}}
- R_2=(T_2=Recipient, L_2=Operation, D_2={"government", "federal tax", "research"}, S_2={"total", "partial"}}.

C. Privacy Assertion

Clients and providers 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_i) on a resource *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. A privacy assertion on *rs* according to this rule may state that *rs* will be shared with government agencies and research institutions. We use the propositional formula "government ∧ research" to specify such statement.

Definition 3 – A *privacy assertion* $A(R_i, rs)$ on *rs* according to R_i=(T_i, L_i, D_i, S_i) is defined by the couple *(pf,g)*; pf = $v_{ip}\wedge\ldots\wedge v_{iq}$ where $v_{ip},\ldots,v_{iq} \in D_i$; $g \in S_i$ is the granularity of *rs* subject to privacy. ◊

D. *Privacy Policy*

Each provider WS has its own perception of what it considers as private. Defining the privacy policy PP^{WS} of WS is performed in two steps. First, the provider identifies the set (noted $\mathcal{P}_p$) of all privacy resources in WS. Second, it specifies assertions for each resource *rs* in $\mathcal{P}_p$. PP^{WS} specifies the way WS (i) treats resources received from C and (ii) *expects* C to treat resources sent by WS. The privacy policy is defined as follows:

Definition 4 – The *privacy policy* of a service is PP^{WS} = $\{A_j(R_i, rs_k)$, j≤$|PP^{WS}|$, i≤$|\mathcal{RS}|$, k≤$|\mathcal{P}_p|$, $rs_k \in \mathcal{P}_p$, $R_i \in \mathcal{RS}\}$. ◊

For instance, let us consider again the previous rule R_1 and R_2. The Lab_Service may specify a set of assertions of each resource to define policy, then

$PP^{Lab\_Service}$=$\{A_1(R_1$,patient_id), $A_5(R_2$,get_lab_result)$\}$, where $A_1(R_1$,patient_id)=("government∧research", total), $A_5(R_2$,get_lab_result())=("federal tax ∧research", total).

E. *Privacy Requirements*

For each Web service WS, client C defines a *Privacy Requirements* $PR^{C/WS}$ stating C's assertions about WS resources. Before creating $PR^{C/WS}$, C first identifies the set (noted $\mathcal{P}_C$) of all privacy resources in WS. $PR^{C/WS}$ assertions describe the following requirements:

- The way C expects the provider to treat the privacy of input data (e.g., patient_id), output data (e.g., experiment results returned by a computational cloud service), and operation usage (e.g., invocation of buy_ingredient());
- The way C treats the privacy of any output data returned by the provider (e.g., test_results).

The requirements are expressed via assertions and express the client's expectations. Client may unequally value the assertions specified in $PR^{C/WS}$ by assigning a weight W_j to each $A(R_i, rs)$ in $PR^{C/WS}$. The higher is the weight, the more important is the corresponding assertion. Each weight is decimal number between 0 and 1. The total of weights assigned to all assertions within equals 1:

- $\forall$j, 1≤j≤$|PR^{C/WS}|$: $0 < W_j \leq 1$.
- $\sum_{j=1}^{k} W_j = 1$, where k=$|PR^{C/WS}|$

When it comes to privacy, clients may be willing to update some of their privacy requirements. To capture this aspect, client C stipulates whether an assertion $A(R_i, rs)$ is mandatory or optional via a boolean attribute M_j attached to A_j. We give below a definition of privacy requirements.

Definition 5 – The *privacy requirements* of a service C is defined as $PR^{C/WS}$=$\{(A_j(R_i, rs_k), W_j, M_j)$, j≤$|PR^{C/WS}|$, i≤$|\mathcal{RS}|$, k≤$|\mathcal{P}_c|$, $rs_k \in \mathcal{P}_c$, $R_i \in \mathcal{RS}$, W_j is the weight of A_j, M_j=True iff A_j is mandatory$\}$. ◊

III. Privacy Compatibility

In this section, we first define the notion of privacy subsumption. Then, we present our cost model-based privacy matching technique.

A. *Privacy Subsumption*

Defining an assertion $A(R_i, rs)$=(pf, g) for *rs* involves assigning value(s) from D_i to the propositional formula *pf* of *A*. The values in D_i are related to each other. For instance, let us consider the domain D_i={"public", "government", "federal tax", "research"} for a rule dealing with the recipient topic. The value public is more general than each other value in D_i. To capture the semantic relationship among domain values, we introduce the notion of *privacy subsumption* (noted $\sqsubseteq$).

Definition 6 – Let D_i = $\{v_{i1},\ldots,v_{im}\}$ be the domain of a rule R_i. We say that v_{ip} is *subsumed by* v_{iq} or v_{iq} *subsumes* v_{ip}, (1≤p≤m and 1≤q≤m) noted $v_{ip} \sqsubseteq v_{iq}$, iff v_{iq} is more general than v_{ip}. ◊

We generalize the notion of privacy subsumption to assertions. 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, if *pf* is true, then *pf'* should be true as well.

Definition 7 – Let us consider $A(R_i, rs)$=(pf, g) and $A'(R_i', rs)$=(pf', g'). *A'* is *subsumed by A* or *A* subsumes *A'*, noted $A' \sqsubseteq A$, if R_i=R_i', rs=rs', *g*=*g'*, and $pf \Rightarrow pf'$. ◊

B. *Matching Privacy Requirements and Policies*

Before client C and service WS start interacting with each other, it is important to verify the compatibility of $PR^{C/WS}$ and PP^{WS}. This task is performed by *Privacy Compatibility Matching* (*PCM*) module. The aim of PCM is to check that assertions in $PR^{C/WS}$ and PP^{WS} are related via subsumption relationships. Two options are possible while matching $PR^{C/WS}$ and PP^{WS}. The first option is to require full matching. This is not flexible since some clients may be willing to use a service even if certain of their privacy constraints are not satisfied. 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*. It is not always possible to find PP^{WS} that fully matches a client's $PR^{C/WS}$. The *privacy matching degree* gives an estimate about the ratio of $PR^{C/WS}$ assertions that are matched to PP^{WS} assertions. We refer to $\mathcal{M} \subset PR^{C/WS}$ as the set of all such $PR^{C/WS}$ assertions. The degree is obtained by adding the weights of all assertions in $\mathcal{M}$:

- Degree ($PR^{C/WS}$, PP^{WS})= $\sum W_j$ for all assertions ($A_j(R_i, rs_k)$,W_j, M_j) $\in \mathcal{M}$.

The threshold τ is provided by a service client and illustrates the minimum value allowed for a matching degree. We give clients the possibility to control their "core" privacy requirements by associating a mandatory attribute M_j to each assertion ($A_j(R_i, rs_k)$,W_j, $M_j\}$ in $PR^{C/WS}$. The PCM determines that $PR^{C/WS}$ and PP^{WS} are compatible if the following holds:

- The privacy matching degree is above the threshold set by C: Degree ($PR^{C/WS}$,PP^{WS}) ≥ τ.
- Every non-matched $PR^{C/WS}$ assertion is optional: $\forall$ ($A_j(R_i, rs_k)$, W_j, M_j) $\in$ ($PR^{C/WS}$-$\mathscr{M}$): M_j = “False”.

IV. Privacy Requirements-aware Adaptation

The recent efforts have highlighted the potential to improve performance by introducing methods to personalize services based on individual information. For instance, a personal’s location, demographics, and past services invocation may be useful in enhancing the efficiency and quality of service provider [10]. However, as we demonstrated in the previous section, the input data collection by service provider impacts the compatibility of PP and PR. The *Privacy Compatibility Matching* (*PCM*) algorithm checks whether $PR^{C/WS}$ is compatible with PP^{WS}. If not, both C and WS are informed by PCM about the assertions in $PR^{C/WS}$ that are incompatible with the assertions in PP^{WS} and may be negotiable (cf. Figure 2).

A. Requirements-Utility cost Model

The *Privacy Compatibility Matching* (*PCM*) module checks whether $PR^{C/WS}$ is compatible with PP^{WS}. If not, both C and WS are informed by PCM about the assertions in $PR^{C/WS}$ that are incompatible with the assertions in PP^{WS} and may be negotiable. In this case, WS starts negotiating with C. The negotiation process is guided by incentives offered by WS to C. Indeed, C may be willing to change its current $PR^{C/WS}$ if certain incentives are provided by WS. The study in [9] shows that a significant percentage of clients are willing to provide additional data if providers offer incentives. Each offer carries an incentive; we assume the existence of a domain-dependent *incentive ontology* that represents the set of possible incentives. An example of incentive in business is “discount”; an example of incentive in Web is “faster response time”.

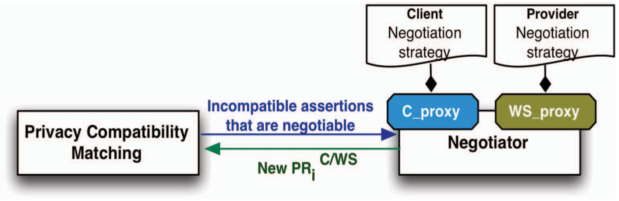

Client Negotiation strategy
Provider Negotiation strategy
C_proxy
WS_proxy
Privacy Compatibility Matching
Incompatible assertions that are negotiable
New $PR_i^{C/WS}$
Negotiator

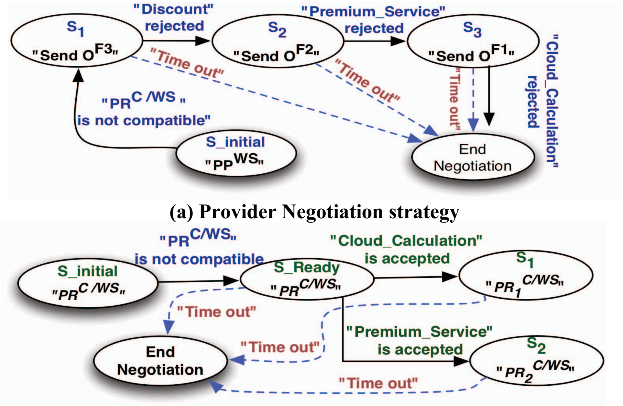
Figure 2. The Negotiation Process

Each client and provider defines its own negotiation strategy beforehand. The *Negotiator* (Figure 2) handles the negotiation process by comparing the client’s and provider’s strategies according to a negotiation protocol. If an offer is accepted, C updates its current privacy requirements. The new requirements are then checked again by PCM for compatibility with PP^{WS}. A successful negotiation concludes with a mutually agreed and signed contract, called privacy e-contract. Moreover, we have to note that the compatible PR and PP of related services that would be interacting are signed through an e-agreement. In [4] we presented an advanced approach to deal with this issue.

B. Defining Negotiation Strategies

WS initially defines a finite set of offers $Ofr = \{O^{F1},\ldots,O^{Fn}\}$. Each O^{Fj} (with $1 \le j \le n$) is transmitted to C until an accepted offer is reached or WS has no further offers to send. The ranking of offers to be sent is illustrated according a *negotiation strategy*. The strategy is described as a state machine where each state represents O^{Fj}; a transition between states represents either an “accept” or “reject” response from C. For instance let us consider the incentives “Cloud_Calculation”, “Premium_Service”, and “Discount”. Figure 4.a depicts an example of WS negotiation strategy. Offers O^{F3}, O^{F2}, and O^{F1} correspond to states: S_1, S_2, and S_3, respectively. The transitions (S_1_S_2), (S_2_S_3), and (S_3_End Negotiation) mean respectively that the three offers were not accepted. Timeout guards are used to end negotiation if a response is not received from C.

On the other side, C defines a set of alternative privacy requirements PR=$\{PR_1^{C/WS},\ldots,PR_n^{C/WS}\}$. C’s negotiation strategy is also described as a state machine; each state represents a requirement $PR_i^{C/WS}$ in PR; a transition corresponds to an incentive accepted by C. Acceptance of an incentive results into a new $PR_k^{C/WS}$ that replaces C’s previous requirement. Figure 3.b depicts an example of a client’s strategy. It shows that C is first interested in the incentive “Cloud_Calculation”, then “Premium_Service”. C associates $PR_1^{C/WS}$ and $PR_2^{C/WS}$ to “Cloud_Calculation” and “Premium_Service”, respectively. When a negotiation begins, C’s negotiation process moves to the S_Ready state. If C receives one of the previous incentives, it adapts $PR^{C/WS}$ to $PR_1^{C/WS}$ (i.e., moves to S_1) or $PR_2^{C/WS}$ (i.e., moves to S_2).



(a) Provider Negotiation strategy

(b) Client Negotiation strategy

Figure 3. Examples of Negotiation Strategies

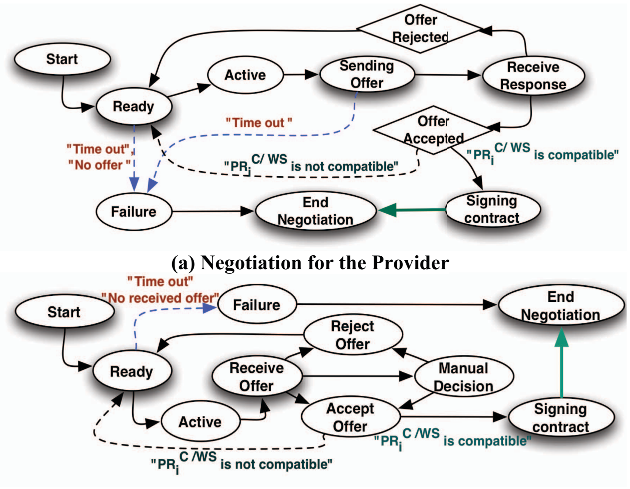
C. Negotiation Protocol

The negotiation protocol describes the sequence of actions performed during a negotiation process. The negotiator creates two proxies *WS_proxy* and *C_proxy* that act on behalf of WS and C, respectively. The negotiator plays the role of coordinator between WS_proxy and C_proxy; it handles the passing of negotiation terms between the two proxies. The Negotiator is a trusted party; neither C nor WS is able to know about the strategy of the other entity.

Figure 4.a shows the negotiation process from the provider's perspective. WS_proxy initiates negotiation by sending O^{Fj} to C_proxy. If O^{Fj} is accepted, C_proxy submits a new $PR_i^{C/WS}$. PCM then checks compatibility of $PR_i^{C/WS}$ and PP^{WS}.

In case of compatibility, both proxies sign a privacy e-contract. Otherwise, WS_proxy sends the next offer to C_proxy. The same process is repeated until an offer is accepted by C_proxy or WS_proxy has no further offers to send. Figure 4.b shows the negotiation process from the client's perspective. The C_proxy evaluates each received offer according to its negotiation strategy. C_proxy may adjust its $PR^{C/WS}$ to have $PR_i^{C/WS}$ if a received offer is accepted. Additionally C_proxy could enforce hands over control to the client in order to reject or accept an offer. This is represented by the "manual decision" in Figure 4.b.

Figure 5 shows the sequence diagram of an example of negotiation process. Since $PR_1^{C/WS}$, which corresponds to the acceptance of "Premium_Service" offer, is incompatible with PP^{WS}, WS_proxy sends the "Cloud_Calculation" offer. C_proxy returns $PR_2^{C/WS}$. Since $PR_2^{C/WS}$ and PP^{WS} are compatible, a privacy e-contract is signed by C_proxy and WS_proxy.



(a) Negotiation for the Provider

(b) Negotiation for the Client

Figure 4. Negotiation Actions

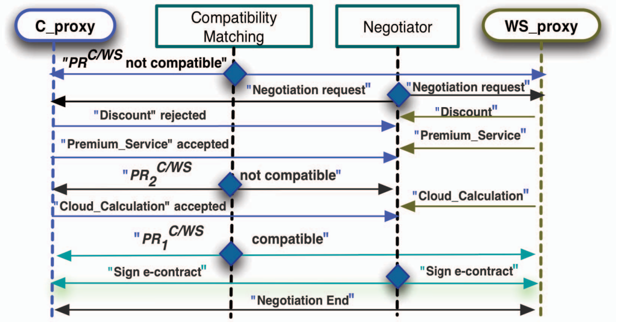


Figure 5. Negotiation process sequence-diagram

V. Prototype Implementation

To illustrate the viability of our approach, we implemented a Meerkat prototype. The prototype architecture is shown in Figure 6.

The *Administrator Interface* allows Meerkat administrators to create privacy rules, privacy subsumption relationships, and the incentive ontology. The *Provider* and *Client Interfaces* enable providers and clients to express their privacy policies and requirements as well as their negotiation strategies. The *Privacy Definition* component manages Meerkat artifacts such as privacy policies, requirements, rules, subsumptions, and incentives. Privacy rules, subsumptions, and incentives are stored in a MySQL database. Privacy policies and requirements are generated in XML format. The *Privacy Compatibility Matching* component implements the matching technique described in section III. It uses a theorem prover to check assertion subsumptions. The *Negotiator* implements the negotiation protocol presented in section IV. We use *SCXML* standard [6] to specify and process negotiation strategies. The prototype is implemented in Java. The user interfaces portion of the prototype is implemented using Java Server Pages (JSP). JSP allows Meerkat's application server (Tomcat), to quickly generate HTML Web pages dynamically based on the actions of Java Servlets. Figure 7 shows an example privacy policy screenshot (part a) and an example of negotiation screenshot (part b) displayed by Meerkat.

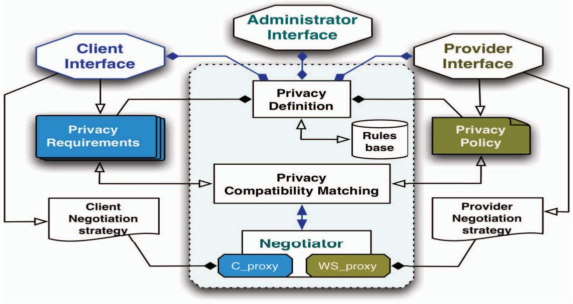
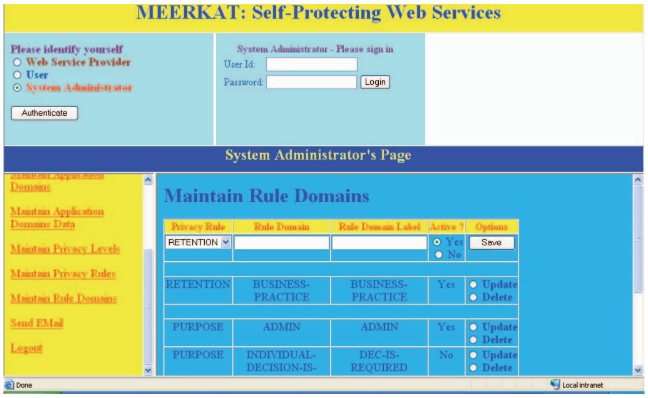


Figure 6. The Meerkat Prototype Architecture



MEERKAT: Self-Protecting Web Services

Please identify yourself
○ Web Service Provider
○ User
● System Administrator

Authenticate

System Administrator - Please sign in
User Id:
Password: Login

System Administrator's Page

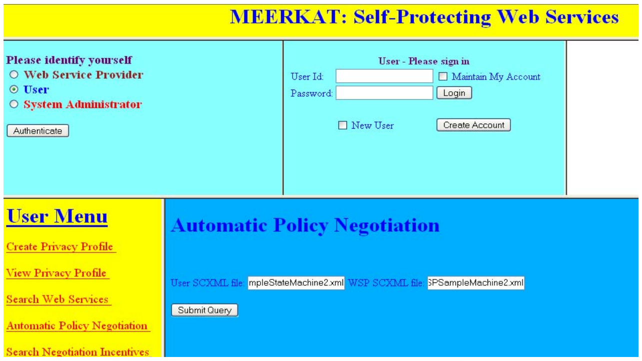
Maintain Application Domains
Maintain Application Domains Data
Maintain Privacy Levels
Maintain Privacy Rules
Maintain Rule Domains
Send EMail
Logout

Maintain Rule Domains

Privacy Rule	Rule Domain	Rule Domain Label	Active ?	Options
RETENTION			● Yes ○ No	Save
RETENTION	BUSINESS-PRACTICE	BUSINESS-PRACTICE	Yes	○ Update ○ Delete
PURPOSE	ADMIN	ADMIN	Yes	○ Update ○ Delete
PURPOSE	INDIVIDUAL-DECISION-IS-	DEC-IS-REQUIRED	No	○ Update ○ Delete

Done
Local intranet

(a) Interface of Privacy Rule Specification



Please identify yourself
○ Web Service Provider
● User
○ System Administrator

User SCXML file: mpleStateMachine2.xml WSP SCXML file: SPSampleMachine2.xml

(b) Setup Automated Negotiation

Figure 7. Meerkat Prototype Interfaces

VI. Related Work

Several approaches have been proposed to preserve privacy in Web services in the context of negotiation. An extension of P3P is proposed in [2]. 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 [3] aims at accomplishing privacy-aware access control by adding negotiation protocol and encrypting data under the classified level. Another effort has been proposed in [10] on the feasibility of achieving a balance between clients' privacy and provider search quality. An algorithm is provided to the client for collecting, summarizing, and organizing their personal information into a hierarchical profile. Through this profile, the client controls which portion of its private information is exposed to the provider by adjusting a threshold. Some policy languages, such as XACML[12], ExPDT[11] 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 negotiation mechanism when incompatibility occurs. In [8], privacy only takes into account a limited set of data fields and rights. The provider specifies the mandatory and optional data for querying the service; the user specifies the type of access for each part of his personal data contained in the service using a DAML-S ontology.

In contrast to the existing approaches, Meerkat introduce a service-oriented privacy model for Web services that goes beyond "traditional" data-oriented privacy approaches. Input/output data as well as operation invocation may reveal sensitive information about clients and hence, should be subject to privacy constraints. In addition, privacy compatibility checking is based on a dynamic client-defined cost model. Finally, Meerkat negotiation model allows providers and clients to automatically negotiate their privacy definitions.

VII. Conclusion

In this paper, we proposed a formal privacy for Web services which deals with privacy at two different levels: data (inputs and outputs) and operation. Clients and providers specify their privacy concerns via privacy requirements and policies, respectively. Both privacy requirements and policies refer rules that may be added, deleted, and modified at any time. We introduced a cost model-based protocol for checking the compatibility of privacy requirements and policies. We introduced a negotiation model that dynamically reconciles requirements with policies in case of incompatibility between clients' and providers' privacy definitions. As future work, we plan to extend our privacy model to take into account the adversary attacks against privacy. Particularly, we will consider the context f service composition and how assure the privacy between services within a composition. We will intend also to extend the negotiation approach in way that both clients and providers can make offers.

Acknowledgment

This work is partially supported by the French National Research Agency (ANR) under the grant number ANR-09-SEGI-008.

References

[1] S-E. Tbahriti, M. Mrissa, B. Medjahed, C. Ghedira, M. Barhamgi, and J. Fayn: "Privacy-Aware DaaS Services Composition". In DXA 2011: pp. 202-216.

[2] O. Kwon, "pervasive P3P-based negotiation mechanism for privacy-aware pervasive e-commerce", In Jouranl of Decision Support Systems, vol. 50, 2010, pp. 213-221.

[3] H.-A Park, J. Zhan and D.H. Lee, "Privacy-Aware Access Control through Negotiation in Daily Life Service". Proceeding of the IEEE ISI workshops, 2008, pp. 514-519.

[4] H-L., S. Dustdar, , J. Goetze, T. Fleuren, P. Mueller, S-E Tbahriti, M. Mrissa, C. Ghedira, "Exchanging Data Agreements in the DaaS Model" In APSCC, 2011 (to be appears).

[5] L. Motiwalla and X. Li, "Value Added Privacy Services for Healthcare Data" In World Congress on Services, SERVICES 2010.

[6] State Chart XML (SCXML): State Machine Notation for Control Abstraction, 2010, http://www.w3.org/TR/scxml/

[7] M. Mrissa, S-E. Tbahriti, and H-L. Truong, "Privacy Model and Annotation for DaaS" In ECOWS 2010, pp. 3-10.

[8] A. Tumer, A. Dogac, and I. H. Toroslu, "A semantic- based user privacy protection framework for web services," in ITWP, vol. 3169. Springer, 2003, pp. 289–305.

[9] R. Druecke, "Attitudes to Privacy at using mobile phones" (in german), Technical Report, Mobile Internet Business, No. 3, 2006, ISSN 1861-3926.

[10] Y. Xu, B. Zhang, and K. Wang, Privacy-enhancing personalized web search, WWW, 2007.

[11] S. Sackmann and M. Kähmer, "ExPDT: A policy-based approach for automating compliance". Wirtschaftsinformatik, 50(5):366–374, 2008.

[12] OASIS. Extensible access control markup language. http://www.oasis-open. org/committees/xacml/, 2008.

[13] M. Barhamgi, D. Benslimane, C. Ghedira, S-E. Tbahriti, M. Mrissa, "A Framework for Building Privacy-Conscious DaaS Service Mashups". ICWS 2011, pp. 323-330.