2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology

Meerkat – A Dynamic Privacy Framework for Web Services

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

[1] University de Lyon
LIRIS-CNRS
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

Abstract— **In this paper, we present Meerkat, a dynamic framework 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. Finally, we propose a negotiation model to reconcile clients' requirements with providers' policies in case of incompatibility.**

Keywords: Privacy, Web services, Negotiation.

I. INTRODUCTION

Privacy is the right of an entity to determine when, how, and to what extent it will release private information [1]. 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). 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/usage. 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. For instance, let us consider a biologist submitting drug discovery data to an Infrastructure-as-a-Service (IaaS)[4]. The Web service runs computationally-intensive experiments and returns results to the biologist. The biologist considers both inputs and outputs as confidential as they may reveal important information to competitors. Finally, clients may view their operation invocations (independently of input/output data) as sensitive. For example, let us consider a patient that invokes 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.

In a service-based system, each client/provider specifies how it handles private information, and how it expects the other entity to treat that information. A provider WS has a privacy policy PP^{WS} that specifies the set of privacy practices 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 $PR^{C/WS}$. For instance, C's privacy requirement about last_name may be tighter its requirement for zip_code. 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 dynamic framework, called Meerkat, for privacy-preserving Web service interactions. The paper's contribution include a Web service-aware privacy model, a cost model-based privacy matching protocol, and a negotiation model based on incentives to reconcile privacy requirements and policies.

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. We provide concluding remarks in Section V.

II. PRIVACY MODEL

The goal of a privacy-preserving framework is to protect private information. We refer to such information as *privacy resources* (simply *resources*). In a service-oriented setting, different types of information may be subject to privacy [2]. For instance, a client may consider an input parameter given to a provider as private; another may view the information stating that the client invoked a specific operation of a given provider as private. To take into account the type of resources, we introduce the notion of *privacy level*

A. Privacy Level

We define two privacy levels in Meerkat: *data* and *operation*. The *data level* deals with the privacy of data shared between clients and providers. Data resources refer to the input and output parameters of a service operation (e.g., defined in WSDL). The *operation level* copes with the privacy of operation usage/invocation. Information about operation invocations is usually stored in a *service log* at the provider's side. Such information includes the invoked operation (what was invoked?), client ID (who invoked it?), invocation date/time (when?), and outcome (how was the outcome?). The outcome states whether the provider has successfully performed the operation or not. The client ID identifies the client such as a company name, company tax ID or DUNS number, service ID, person's national identifier, and so on. Similarly to input/output data, service log entries may be subject to privacy concerns [6].

B. Privacy Rule

The sensitivity of a resource may be defined according to several dimensions called *privacy rules*. We define a privacy rule by a *topic*, *level*, *domain*, and *scope*. The *topic* gives the privacy facet represented by the rule. For instance, the "purpose" topic states the intent for which a resource collected by a provider will be used. The *level* represents the privacy level of all resources on which the rule is applicable. Each rule has one single level: "data" or "operation". We use the terms *data* and *operation rule* to refer to a rule with a "data" and "operation" level, respectively. The *domain* is a finite set that enumerates the possible privacy values that can be taken by resources according to the rule's topic. For instance, the retention topic is {"no-retention", "indefinitely", "stated-purpose"}. The *scope* of a rule defines the granularity of the resource that is subject to privacy constraints. We consider two cases: operation and data rules. In the former case, several parts of a service log entry may be viewed as private. 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, then the whole entry of that operation in the service log is private. Otherwise (i.e., the assigned scope is "partial"), only the ID of the client that invoked the operation 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".

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 called *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. For instance, *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*. For example, let us consider a rule for the topic "recipient" with a domain equal to {"local", "government", "public", "research"}. 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 *pf* = "government $\wedge$ research" to specify such statement.

D. Privacy Policy

A service WS has a *privacy policy* that specifies the set of practices applicable to WS clients. 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, the provider specifies assertions for each resource *rs* in $\mathcal{P}_p$. Deciding about the content of $\mathcal{P}_p$ and the rules to apply to each resource in $\mathcal{P}_p$ varies from a provider to another.

During an interaction between WS and a client C, WS receives information from C and returns information (i.e., results) to C. PP^{WS} specifies the way WS (i) treats resources received from C and (ii) *expects* C to treat resources sent by WS. We consider three cases: (a) *rs* is an input data, (b) *rs* is an output data, and (c) *rs* is an operation. If *rs* is an input data or operation (cases (a) and (c)), then $A(R_i,rs)$ states what will the provider do with *rs* according to R_i. If *rs* is an output data (case (b)), then WS defines two assertions for *rs* according to R_i; the first assertion, noted $A(R_i,rs^E)$, gives WS expectation; the second assertion, $A(R_i,rs^P)$, denotes WS practice:

- *Expectation*: $A(R_i,rs^E)$ states what the provider *expects* the client to do with *rs* according to R_i.
- *Practice*: $A(R_i,rs^P)$ states what the provider will do with *rs* according to R_i. For instance, let us consider a scientist that would like to conduct some experiments. Since the experiments are time-consuming, the scientist uses a powerful cloud service. To assure the scientist about the confidentiality of the experiment results, the provider declares its practices regarding the privacy of the returned results.

E. Privacy Requirements

Clients may expect or require different levels of privacy according to their perception of information sensitivity [5]. The client's viewpoint about privacy depends not only on the resources (e.g., date_of_birth, last_name) but also on the services to be used. 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, output data (e.g., experiment results returned by a computational cloud service), and operation usage; and
- The way C treats the privacy of any output data returned by the provider.

The aforementioned requirements are expressed via privacy assertions. Similarly to privacy policies, requirements on outputs express the client's expectations (noted $A(R_i,rs^E)$) and practices (noted $A(R_i,rs^P)$).

Client C may unequally value the assertions specified in $PR^{C/WS}$. For instance, C's requirements about social_security_number (national identifier used in the US) may be stronger than its requirements for zip_code. Besides, C may consider an assertion more essential than another, even if both assertions are about the same resource. For example, C may view the rule constraining the recipients of social_security_number as more valuable than the rule stating the duration for which the service can retain social_security_number. For that purpose, C assigns a weight W_j to each assertion $A(R_i,rs)$ in $PR^{C/WS}$. W_j is an estimate of the significance of $A(R_i,rs)$ from the client's point of view. 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:

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

When it comes to privacy, clients may be willing to change some of their privacy requirements. For instance, shoppers may agree to relax constraints about the divulgence of their zip code if the provider offers incentives such as discounts and coupons. However, the same shoppers will probably be more reluctant to loosen conditions about the divulgence of their names. 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.

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 a resource *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 {"public", "government", "federal tax", "research"} for a rule dealing with the recipient topic (i.e., T_i = "recipient"). The value public is more general than each other value in D_i. Indeed, if the recipient of *rs* is declared public (i.e., shared with any entity), then the recipient is also government, federal tax, and research. 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; federal tax $\sqsubseteq$ public; research $\sqsubseteq$ public; federal tax $\sqsubseteq$ government. Privacy subsumption is transitive since it models the "is-a" relationship. We use $\sqsubseteq^*$ to refer to the transitive closure of $\sqsubseteq$.

We generalize the notion of privacy subsumption to assertions. Let us consider an assertion $A(R_i,rs)=(pf,g)$ representing an expectation of a client C (resp., provider WS) and another assertion $A'(R_i',rs')=(pf',g')$ modeling a practice of a provider WS (resp., client C). 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 C (resp., WS) as stated by *pf* should be more general (i.e., subsumes) than the practice of WS (resp., C) as given by *pf'*. In other words, if *pf* is true, then *pf'* should be true as well. For instance, if pf_k = government ∧ research and *pf'*= government, then $pf \Rightarrow pf'$ (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 *pf* are syntactically different from the ones used in *pf'*, they may be semantically related via subsumption relationships. For instance, let us assume that *pf* = government ∧ research and *pf'* = federal tax. Since federal tax $\sqsubseteq$ government, we can state that government $\Rightarrow$ federal tax. In this case, we can prove that $pf \Rightarrow pf'$ and hence, $A' \sqsubseteq A$. To deal with the issue of having different literals in propositional formulas, we use the following property: if $v_{ip} \sqsubseteq^* v_{iq}$ (i.e., v_{iq} directly or indirectly subsumes v_{ip}), then $v_{iq} \Rightarrow v_{ip}$.

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 Meerkat's *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 (cf. Definition 7). As mentioned in Section II.D and II.E, both $PR^{C/WS}$ and PP^{WS} contain expectations (i.e., requirements) and practices. PCM matches expectations in $PR^{C/WS}$ to practices in PP^{WS} and expectations in PP^{WS} to practices in $PR^{C/WS}$. PCM algorithm deals with the following three cases:

- Case (a) – PCM matches a $PR^{C/WS}$ assertion $A(R_i,rs)$ where rs is an input or operation usage, to an assertion $A'(R_i',rs')$ in PP^{WS}. In this case, $A(R_i,rs)$ is a C's expectation and $A'(R_i',rs')$ is a PP^{WS} practice. If $A' \sqsubseteq A$ then A' and A are matched.
- Case (b) – PCM matches a $PR^{C/WS}$ assertion $A(R_i,rs^E)$ where rs^E is an output, to an assertion $A'(R_i',rs^{P}')$ in PP^{WS}. In this case, $A(R_i,rs^E)$ is a C's expectation and $A'(R_i',rs^{P}')$ is a PP^{WS} practice. If $A' \sqsubseteq A$ then A' and A_k are matched.
- Case (c) – PCM matches a $PR^{C/WS}$ assertion $A(R_i,rs^P)$ where rs^P is an output, to an assertion $A'(R_i',rs^{E}')$ in PP^{WS}. In this case, $A(R_i,rs^P)$ is a C's practice and $A'(R_i',rs^{E}')$ is a PP^{WS} expectation. If $A \sqsubseteq A'$ then A' and A are matched.

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*. Due to the large number and heterogeneity of Web services, it is not always possible to find policy PP^{WS} that fully matches a client's requirement $PR^{C/WS}$. The *privacy matching degree* is the sum of the weights of $PR^{C/WS}$ assertions that are matched to PP^{WS} assertions. The *privacy matching threshold* τ gives the minimum value allowed for a matching degree. The value of τ is given by the client and gives an estimate of how much privacy the client is willing to "sacrifice". The PCM determines that $PR^{C/WS}$ and PP^{WS} are compatible if (i) The privacy matching degree is above the threshold set by C and (ii) Every non-matched $PR^{C/WS}$ assertion is optional.

IV. Meerkat Negotiation Model

The *PCM* 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 becomes aware of the dollar-value of its $PR^{C/WS}$ and may be willing to change its current $PR^{C/WS}$ if certain incentives are provided by WS [3]. Studies showed 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".

Each client and provider defines its own negotiation strategy beforehand. The *Negotiator* 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. Defining Negotiation Strategies

WS initially defines a finite set of offers $Ofr = \{O^{F1},...,O^{Fn}\}$. Each O^{Fj} (with $1 \leq j \leq n$) 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.

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.

B. 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. 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 5.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.

V. Conclusion

In this paper, we proposed a dynamic privacy framework, called Meerkat, for Web services. Meerkat deals with privacy at two different levels: data and operation. Clients and providers specify their privacy concerns/practices via privacy requirements and policies, respectively. 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 are planning to extend the negotiation model so that both clients and providers can make offers. We will also adapt the techniques proposed in this paper to provide for privacy-preserving service composition.

References

[1] B. C. M. Fung, K. Wang, R. Chen and P. S. Yu, "Privacy-Preserving Data Publishing: A Survey of Recent Developments", In Journal of ACM Computing Surveys, 2010. vol. 42, no. 4, pp 1-53.

[2] Y. Xu, K. Wang, B. Zhang, Z. Chen, "Privacy-enhancing personalized web search", In Proceeding of the International Conference on World Wide Web, 2007, pp. 591-600.

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

[4] K. Keahey, M. Tsugawa, A. Matsunaga, J. Fortes, "Sky Computing", In Journal of IEEE Internet Computing, 2009. vol. 13, no.5, pp. 43-51.

[5] L. Motiwalla and X. Li, "Value Added Privacy Services for Healthcare Data", In Proceeding of 6th World Congress on Services, SERVICES 2010.

[6] J. Kawamoto and M. Yoshikawa, "Security of social information from query analysis in DaaS", In Proceedings of the EDBT/ICDT Workshops, EDBT/ICDT '09, pp. 148-152, New York, USA, 2009.