# A Situation-aware Access Control based Privacy-Preserving Service Matchmaking Approach for Service-Oriented Architecture

Stephen S. Yau and Junwei Liu
Arizona State University
Tempe, AZ 85287-8809, USA
{*yau, junwei.liu*}*@asu.edu*

## Abstract

*Service matchmaking is an important process in the operation of Service-Oriented Architecture (SOA) based systems. In this process, information from both service providers and requestors are used. How to protect the privacy of participating parties during the matchmaking process imposes a challenge. In this paper, a privacy-preserving service matchmaking approach is presented to support semantic-based service matchmaking and avoid privacy leakages to untrusted parties. The approach uses situation-aware access control (SA-AC) mechanism to ensure the appropriate disclosure and use of private information by modeling, specifying and enforcing SA-AC policies. It provides an owner-centric mechanism for both service providers and requestors in SOA-based systems to protect their private information during service matchmaking.*
***Keyword****: Service-oriented architecture, SOA-based systems, service matchmaking, privacy-preserving, situation-aware access control.*

## 1 Introduction

Service-Oriented Architecture (SOA) enables rapid composition of distributed applications from services in a flexible and agile manner [1]. A *service* is a well-defined and self-contained software entity with a discoverable and invocable interface to provide certain capability over networks using standard protocols. Services developed using different programming languages and deployed over various service platforms can seamlessly interoperate across multiple domains over heterogeneous networks. Service matchmaking enables service requestors to locate the most suitable services among a large number of available services with specific and personalized requirements.

Traditional service matchmaking approaches [2–7] provide coarse service matching based on syntactical matching of service names and properties. The lack of consideration of service semantics imposes great difficulties when move to automatic semantic-based service discovery exploiting the Semantic Web. With the development of semantic service specification languages, semantic-based service matchmaking approaches based on service parameters [8], functionalities [9] and related contextual data [10] have been developed. A simple example scenario of semantic-based service matchmaking is to find a nearest drug store to buy and pickup a particular medicine considering the current location of the requestor, store hours and the accepted payment methods of the store.

Such a semantic-based service discovery process requires more semantic information about the services, service providers and service requestors to be processed during service matchmaking. With the increasing concern on privacy, service matchmaking approaches should provide appropriate mechanisms to protect the private information of parties involved. For example, in the above scenario, the address and medicine information of the service requestor should not be divulged to untrusted parties. How to protect the privacy during service matchmaking imposes a challenge. Such requirements for privacy protection become even more stringent when the cross-domain and open characteristics of SOA-based systems (SBSs) are considered.

In this paper, we will present a situation-aware access control based privacy-preserving service matchmaking approach to address this challenge. Our approach uses a trusted third party to host the situation-aware access control (SA-AC) mechanism for flexible access controlling of private information from both service providers and service requestors. Our approach can be incorporated in directory-based service discovery approaches. It provides an owner-centric privacy protection mechanism by allowing information owners to specify access control policies for the private information in service advertisements and requests based on SA-AC policy ontology. Such policies are then enforced to ensure the validated disclosure and use of protected private information.

2007 IEEE International Conference on Web Services (ICWS 2007)
0-7695-2924-0/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

## 2 Current state of the art

Many Service Discovery Protocols (SDPs) have been developed for SBSs. Among them, Jini [2], SLP [3], Salutation [4], UPnP [5], and Bluetooth [6] are five major approaches. As summarized in [11], none of these SDPs address the privacy issues during service matchmaking. The Universal Description, Discovery and Integration (UDDI) [7] project is an industry initiative of service discovery and its registries are open for browsing and querying for services, and hence it also does not address the privacy issues.

Carminati, et al [12] discussed the privacy issues in service discovery agencies and three possible types of solutions. They discussed privacy concerns about untrusted service discovery and leakage of service providers' privacy to untrusted service requestors during service matchmaking process. They also have the concern about leakage of service requestors' privacy during service access phase. In Ninja SDS [13], the privacy of service providers is protected using an access control mechanism, but the privacy of service requestors is not protected. Zhu, et al [14] protected the privacy of both service providers and requestors using owner-based service directories and additional service returning. Their approach requires every entity to have its own directory, which may not be suitable for general SBSs.

There are also approaches addressing other privacy related aspects. Kagal, et al [15] modeled a set of security and privacy ontologies and presented a security specification based matchmaking approach. Platform for Privacy Preferences Project (P3P) [16] provides a standard for services to express their privacy practices, which can be used to reach the privacy agreement during service access phase. The trust negotiation technique [17] focuses on protecting the privacy information during peer-to-peer authentication and trust establishment, which may be used for entities in SBSs to establish trust relations. Anonymous routing techniques, such as ANODR [18], can be used as underlying techniques for anonymous communication between entities during service discovery. These approaches do not consider privacy protection during service matchmaking process.

## 3 Privacy issues in service matchmaking

Privacy-preserving service matchmaking approaches must address three specific privacy issues regarding service matchmaking: untrusted service directory, leakage of service providers' privacy and service requestors' privacy. Figure 1 shows example scenarios for these three privacy issues.

1. **Untrusted service directory**. To perform accurate service matchmaking, service directories have full knowledge of the service advertisements and requests from service providers and requestors, including the private information. However, whether a service directory can be trusted or not is questionable, especially in the open environment of SBSs. An untrusted service directory may collect and abuse the use of private information of both service providers and requestors. For example, in the scenario shown in Figure 1(a), the sensitive parameters and URI description of a salary lookup service and the private identity of a service requestor are exposed to the untrusted service directory.

2. **Privacy of service provider**. Even with a trusted service directory, the private information in service advertisements may be disclosed to untrusted parties. A service requestor may submit tailored requests to the service directories with the intention to look for sensitive information of matched services. For example, in the scenario shown in Figure 1(b), the untrusted service requestor may intentionally look for services whose output is "salary" and S1 will be returned as the matchmaking result, which will result in leaking the sensitive parameter and URI description.

3. **Privacy of service requestor**. Service matchmaking approaches may also support constraints specified by service providers to pre-screen service requestors.

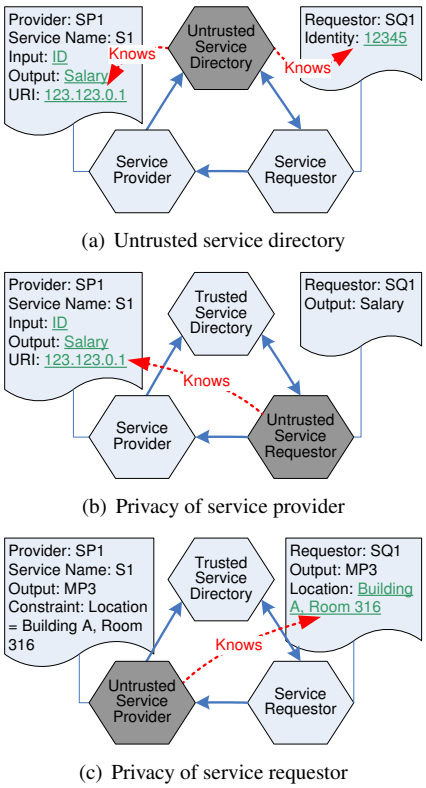


**Figure 1. Privacy leakage scenarios.**

2007 IEEE International Conference on Web Services (ICWS 2007)
0-7695-2924-0/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

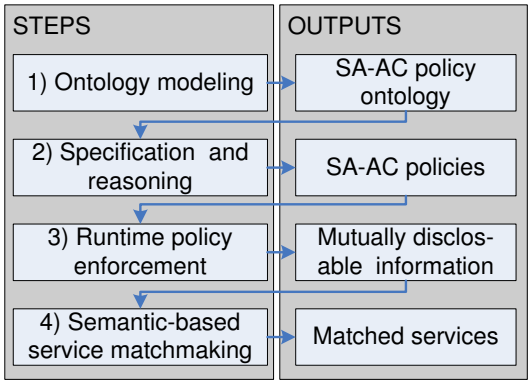


**Figure 2. Overall approach**

However, this may result in leaking service requestors' privacy. This is because service providers can also tailor specific constraints to target specific service requestors. For example, in the scenario shown in Figure 1(c), the untrusted service provider can constrain that only the users in "Building A, Room 316" can discover it, whereas such location information may provide direct evidence of the identity of the service requestors.

To address these privacy issues, the solutions should not only ensure the appropriate submission of private information to service directories, but also ensure the appropriate disclosure and use of private information during service matchmaking with respect to the privacy-preserving preferences of information owners.

## 4. Our SA-AC based privacy-preserving service matchmaking approach

Our privacy-preserving service matchmaking approach utilizes SA-AC mechanism to ensure the appropriate disclosure of private information in both service advertisements and requests based on SA-AC policies specified by information owners. Our approach assumes that there is a trusted third party to host the SA-AC enforcement point, whereas the SA-AC policy enforcement can also be provided by multiple decentralized trusted parties, each in charge of a set of SA-AC policies.

As shown in Figure 2, our approach involves four steps. An OWL-based ontology is first developed to model SA-AC policies with formal logical reasoning support; SA-AC policies are then specified and attached in service advertisements and requests; such policies are enforced at runtime to ensure the appropriate disclosure of private information and finally the semantic-based service matchmaking is performed with the mutual disclosable specifications. In the following subsections, each step will be presented in detail.

### 4.1 OWL-based SA-AC policy modeling

As the first step of our approach, an OWL-based SA-AC policy ontology is developed to model the semantics of SA-AC policies and their relations with situations, services, and entities in SBSs. It serves as the foundation of the knowledge base for multiple parties in SBSs to specify and enforce SA-AC policies. The SA-AC model [19] incorporates situation-aware constraints in RBAC models [20], such that the access control decisions can adapt regarding different situations. Web Ontology Language (OWL) [21] is a W3C standard for representing machine interpretable knowledge in Semantic Web. Figure 3 illustrates the SA-AC policy ontology modeling the entities and relations defined in the SA-AC model and the integration of SA-AC policy ontology with situation ontology [22] and SAW-OWL-S [23] service ontology we have developed. Situation ontology provides a hierarchical modeling approach for context and situation in SBSs and SAW-OWL-S incorporates related contextual and situational information of services into OWL-S ontology.

The SA-AC policy ontology models the following entities and relations for SA-AC policies:

- The user set $\boldsymbol{U}$ is modeled as *Entity* class.
- The object set $\boldsymbol{O}$ is not explicitly modeled and an object can be either a *process* or referred by an *xsd:anyURI*, such as a service process instance "http://dpse.eas.asu.edu/service.owl#process1" or a portion of service profile specification referred by "http://dpse.eas.asu.edu/service.owl#serviceProfile1". Hence, any private information in the OWL-based service advertisements and requests can be denoted by an *xsd:anyURI* and modeled as an object needs to be protected.
- The role set $\boldsymbol{R}$ is modeled as *Role* class and the permission set $\boldsymbol{P}$ is modeled as *Permission* class. These two classes have properties *hasSubRole* and *hasSubPermission* to model the role and permission hierarchies $\boldsymbol{RH}$ and $\boldsymbol{PH}$, respectively.
- The situation set $\boldsymbol{SA}$ is modeled as *Situation* class. A situation can be either an *AtomicSituation* binding a *Entity* with a *SituationAssertion* or a *CompositeSituation* composed by other situations using logical or temporal operators. For example, an instance of SituationAssertion is defined as "Location is in Building A", and a situation instance binds $\text{User}_A$ to this SituationAssertion. Such situation is satisfied if "$\text{User}_A$'s location is in Building A".
- The *SAACPolicy* class has two subclasses: *SAUserRoleAssignment* models situation-aware user role assignments and *SARolePermissionAssignment* models situation-aware role permission assignments.

2007 IEEE International Conference on Web Services (ICWS 2007)
0-7695-2924-0/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

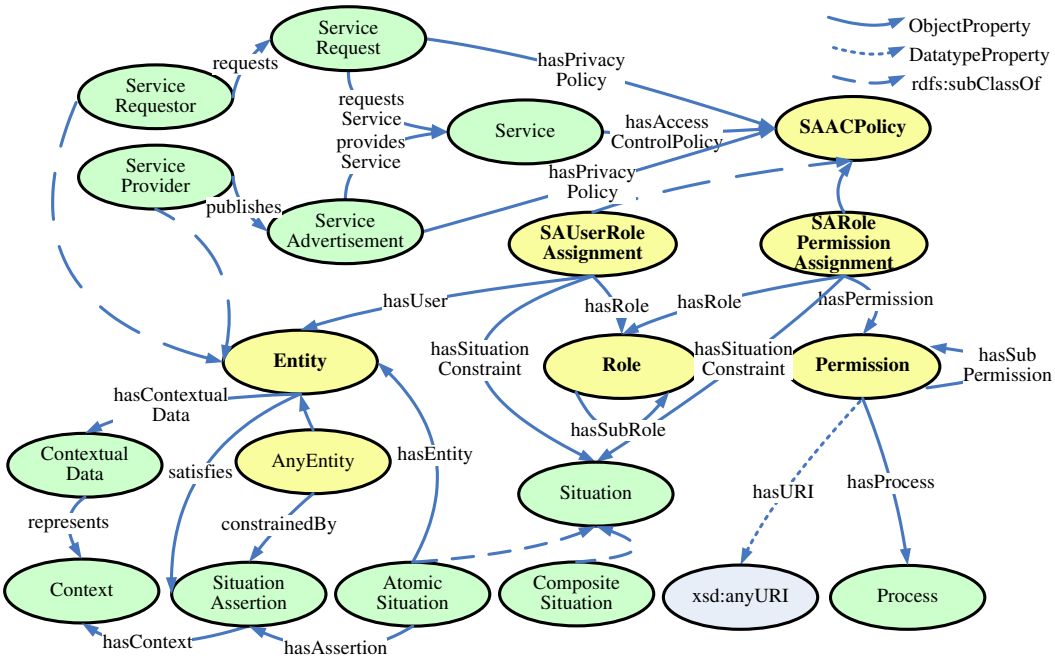


**Figure 3. SA-AC policy ontology**

*SAACPolicy* can serve for two purposes: to control the access to a service during service access phase; or to protect the private information during service matchmaking process as used in our approach. *ServiceProvider* and *ServiceRequestor* can specify owner-centric SA-AC policies for the private inforamtion in *ServiceAdvertisement* and *ServiceRequest*.

It should be noted that our approach not only provides the accessing protection to the whole specification, but also provides flexible accessing protection to any portion of the specification. This gives information owners more flexibility and convenience in privacy protection.

SA-AC model [19] incorporates situation-aware constraints into RBAC model [20] such that the dynamic status of service providers, requestors and environments will all affect the access control decisions. An example of flexible SA-AC policy is "the process model can be disclosed to authenticated users within the domain or any visitor located in the conference room during business hours".

In both SA-AC model [19] and RBAC model [20], the user role assignment is modeled based on specific user identities. Such user role assignment lacks flexibility and faces difficulties in dynamic and open SBSs, in which spontaneous interactions with new users happen frequently and the identities of such new users may not be acquired in advance. Although new policies can be dynamically generated, the arrivals of new users will result in a separate set of policies for each new user and additional overhead introduced by periodical policy generating, updating and enforcing.

To deal with such dynamic and open characteristics of SBSs, a special type of *SAUserRoleAssignment* is modeled in our SA-AC policy ontology to represent dynamic user role assignments based on user contextual data. Instead of a specific *Entity* instance, a flexible contextual data based *SAUserRoleAssignment* is specified with an *AnyEntity* instance constrained by a *SituationAssertion* instance. When enforcing such a policy, any entity whose contextual data satisfying the situation assertion constraint will have the corresponding role. An example of contextual data based user role assignment is "hasUser: AE (constrainedBy: LocationInBuildingA), hasRole: R", which means that any entity located in Building A will have role R. Using such flexible user role assignments, roles can be dynamically assigned to users based on their contextual data at runtime and no knowledge about the users is required in advance. Furthermore, no more policies will be added and no policies will be modified in case any new user appears.

## 4.2 Specification and reasoning of SA-AC policies

In our approach, the service advertisements and requests are specified based on SAW-OWL-S [23], which models the functionality and related contextual data of a service.

In addition to the service specifications, the OWL specifications of SA-AC policies for private protection are also attached in each service advertisement or request. Formal logic reasoning is performed on the OWL-based specifica-


IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

**Table 1. FOL rules for SA-AC policy inference**

| | |
|---|---|
| $\Re 1$: | $\forall$ *saur* $\forall$ *e* $\forall$ *r* $\forall$ *s SAUserRoleAssignment(saur)* $\wedge$ *Entity(e)* $\wedge$ *Role(r)* $\wedge$ *Situation(s)* $\wedge$ *hasUser(saur,u)* $\wedge$ *hasRole(saur,r)* $\wedge$ *hasSituationConstraint(saur,s)* $\wedge$ *satisfied(s)* $\Rightarrow$ *_hasRole(e,r)* |
| $\Re 2$: | $\forall$ saur $\forall$ *e* $\forall$ *ae* $\forall$ *r* $\forall$ *s* $\forall$ *sa SAUserRoleAssignment(saur)* $\wedge$ *Entity(e)* $\wedge$ *AnyEntity(ae)* $\wedge$ *Role(r)* $\wedge$ *Situation(s)* $\wedge$ *SituationAssertion(sa)* $\wedge$ *hasUser(saur,ae)* $\wedge$ *hasRole(saur,r)* $\wedge$ *hasSituationConstraint(saur,s)* $\wedge$ *constrainedBy(ae,sa)* $\wedge$ *satisfies(e,sa)* $\wedge$ *satisfied(s)* $\Rightarrow$ *_hasRole(e,r)* |
| $\Re 3$: | $\forall$ *sarp* $\forall$ *r* $\forall$ *p* $\forall$ *s SARolePermissionAssignment(sarp)* $\wedge$ *Role(r)* $\wedge$ *Permission(p)* $\wedge$ *Situation(s)* $\wedge$ *hasRole(sarp,r)* $\wedge$ *hasPermission(sarp,p)* $\wedge$ *hasSituationConstraint(sarp,s)* $\wedge$ *satisfied(s)* $\Rightarrow$ *_hasPermission(r,p)* |
| $\Re 4$: | $\forall$ *e* $\forall$ $r_1$ $\forall$ $r_2$ *Entity(e)* $\wedge$ *Role*($r_1$) $\wedge$ *Role*($r_2$) $\wedge$ *hasSubRole*($r_1$,$r_2$) $\wedge$ *_hasRole(e,*$r_1$*)* $\Rightarrow$ *_hasRole(e,*$r_2$*)* |
| $\Re 5$: | $\forall$ *r* $\forall$ $p_1$ $\forall$ $p_2$ *Role(r)* $\wedge$ *Permission*($p_1$) $\wedge$ *Permission*($p_2$) $\wedge$ *hasSubPermission*($p_1$,$p_2$) $\wedge$ *_hasPermission(r,*$p_1$*)* $\Rightarrow$ *_hasPermission(r,*$p_2$*)* |
| $\Re 6$: | $\forall$ *e* $\forall$ *r* $\forall$ *p* $\forall$ *uri Entity(e)* $\wedge$ *Role(r)* $\wedge$ *Permission(p)* $\wedge$ *_hasRole(e,r)* $\wedge$ *_hasPermission(r,p)* $\wedge$ *hasURI(p,uri)* $\Rightarrow$ *_canAccess(e,uri)* |

tions for various purposes.

#### 4.2.1 OWL ontology reasoning

Various types of OWL ontology inferences can be performed for analyzing the service and policy specifications, including

- Consistency checking, which checks whether a specification is consistent by reasoning if there is any inconsistent ontology class or inconsistent ontology instance in the specification.
- Implicit knowledge reasoning, which reasons about the implicit knowledge conveyed by the specifications. For example, we can reason implicit knowledge "RoleA hasSubRole RoleC" from explicitly specified knowledge "RoleA hasSubRole RoleB" and "RoleB hasSubRole RoleC" because hasSubRole is a transitive property.

#### 4.2.2 FOL rule-based reasoning

OWL-based specifications support *First-Order Logic (FOL)* rule-based reasoning. To achieve this, the OWL-based specifications are first converted to FOL representations and then FOL rule-based reasoning is performed using FOL provers. The basic idea of converting OWL-based specifications to FOL representations is to translate class references to unary predicates, properties to binary predicates, and OWL axioms (such as *owl:sameAs*) to FOL rules [24].

The FOL rules illustrated in Table 1 are used to reason about the accessing decision of *"whether an entity e can access certain private information referred by uri"*, in which all predicates are automatically converted from the SA-AC policy ontology, except *_hasRole*, *_hasPermission* and *_canAccess*. *_hasRole(e,r)* means entity *e* has role *r*; *_hasPermission(r,p)* means role *r* has permission *p*; and *_canAccess(e,o)* means entity *e* can access object *o*. The situation constraints have been integrated in these three predicates. The FOL rule $\Re 1$ is used for reasoning the situation-aware user role assignments and $\Re 2$ is used for the dynamic user role assignment based on users' contextual data. $\Re 3$ is used for the situation-aware role permission assignments. $\Re 4$ and $\Re 5$ are used for reasoning the hierarchies of roles and permissions. And $\Re 6$ is used for making decision on whether an entity can access an object. Given an SA-AC policy specification, it will be first automatically transformed into FOL representations and then the rules listed in Table 1 are used to make access control decisions automatically. For example, the flexible user role assignment specification $P_1$*(hasUser: AE (constrainedBy: LocationInBuildingA), hasRole: R)* will be transformed to FOL representations *SAUserRoleAssignment(*$P_1$*), AnyEntity(AE), Role(R), Situation(LocationInBuildingA), hasUser(*$P_1$*,AE), hasRole(*$P_1$*,R)* and *hasSituationConstraint(*$P_1$*,LocationInBuildingA)*. It is noted that additional rules will be generated to reason about whether an entity satisfies a situation assertion and whether a situation is satisfied based on the situation ontology specifications [22].

### 4.3 Runtime policy enforcement

The specified SA-AC policies are enforced to ensure the appropriate disclosure and use of protected private information during the service publishing, discovering and matchmaking steps of service discovery process as illustrated in Figure 4. Here, we assume that the communications are secure.

1. During service publish phase, a service provider may specify SA-AC policies which define who under what situation can access what portion of the service advertisement. For SBSs with trusted service directories, such as systems within a centrally administrated domain, the service advertisement can be submitted to the trusted service directories directly. In case the service directory is not fully trusted, the service provider will contact the SA-AC enforcement point enforcing

2007 IEEE International Conference on Web Services (ICWS 2007)
0-7695-2924-0/07 $25.00 © 2007 IEEE

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

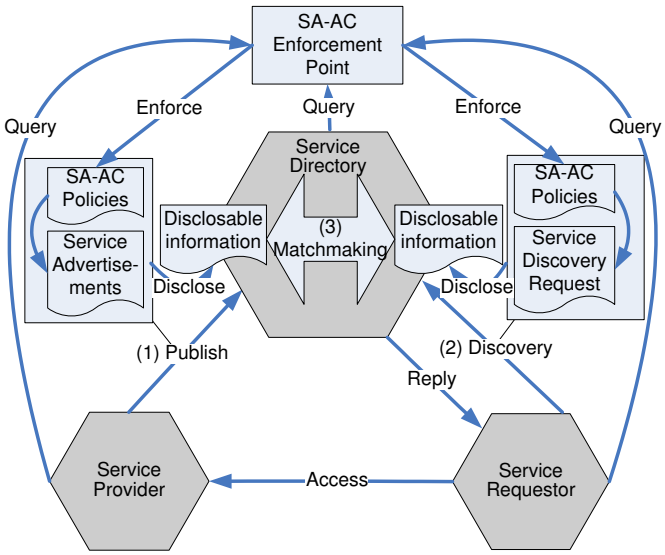


**Figure 4. Runtime SA-AC policy enforcement**

the SA-AC policies to determine what portion of the service advertisement can be published to the service directory. Then, only the disclosable portion will be published to the service directory.

2. Similarly, during service discovery phase, a service requester can also specify SA-AC policies for the private information in service discovery requests. Such policies are enforced to determine what portion of information will be exposed to the service directory.

3. When the service directory receives a service discovery request, before it performs service matchmaking between a registered service and the request, it will first query the SA-AC enforcement point for what information in the involved service advertisement and request can be disclosed to the other party and used for service matching. The SA-AC enforcement point makes decisions based on the SA-AC policies of both the advertisement and request, and the current situations. The service matchmaking is then performed on the mutually disclosable portion of the service advertisement and request.

With such runtime SA-AC policy enforcement, the leakage of privacy to untrusted parties are prevented during service discovery process of SBSs.

On one hand, the leakage of privacy to untrusted service directories is prevented by enforcing SA-AC policies to filter the information before being submitted to service directories. An service directory will not know the existence of any private data which is not disclosed to it. It should be noted that with the information filtering, a service directory may not have the full specification of a service advertisement or request, which may decrease the matchmaking accuracy of the service directory. In extreme cases, the matchmaking will fail due to the lack of necessary information. The formal definition of privacy-preserving strength and how to achieve an appropriate tradeoff between the privacy-preserving strength and accuracy of service matchmaking need to be studies.

On the other hand, the leakage of privacy to untrusted service providers and requestors is prevented by performing service matchmaking only on mutually disclosable portion of service advertisements and requests. In particular, for a service requestor, only the portion of service advertisements which can be accessed by him will be used for matching his request. For any matched service, only the portion of the service advertisement disclosable to him will be returned, Hence, the direct disclosure of private information to untrusted service requestors is prevented. Furthermore, the service requestor cannot deduce any private information which is not disclosable to him from the matchmaking results indirectly, because such information does not participant in the service matchmaking process. The private information of service requestors is also protected from being directly disclosed to or indirectly deduced by untrusted service providers in the same way. It should also be noted that the privacy filtering before service matchmaking may affect the matchmaking accuracy as well. However, as SOA envisions service-rich environments, we can assume that there are services providing similar functionalities with different privacy protection preferences. Thus, the overall matchmaking result is still acceptable.

### 4.4 Semantic-based service matchmaking

As the final step in our approach, the mutually disclosable information from service advertisements and requests is then used for service matchmaking. In our approach, the functionality-based service matchmaking approach presented in [9] is used.

## 5 Discussion

Carminati et al [12] suggested three types of approaches to address the privacy issues related to service discovery agencies: access-control-based approaches, cryptography-based approaches, and hash-based approaches. Access-control-based approaches ensure the appropriate disclosure and use of private information by enforcing access control policies. Our SA-AC based approach falls into this category and is an extension of traditional access-control-based approaches. Cryptography-based approaches protect the private information through information encryption, in which


IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

**Table 2. Comparison of privacy-preserving service matchmaking approaches**

| | SA-AC based | AC based | Cryptography based | Hash based |
|---|---|---|---|---|
| Handles untrusted directory? | Yes | Yes | Yes | Partial |
| Protects provider's privacy? | Yes | Yes | Yes | Partial |
| Protects requestor's privacy? | Yes | Yes | Yes | Partial |
| Supports semantic matching? | Yes | Yes | No | No |
| Handles new entities? | Yes | Partial | Partial | No |
| Efficiency | Low | Low | High | High |

various keys are assigned to users and the encrypted information can only be decrypted by users having the correct keys. Hash-based approaches protect the private information through information hashing, in which the hashed specifications are sent to service directories and service matchmaking is performed on the hashed specifications directly.

We compared our SA-AC based approach with the traditional access-control-based, cryptographic-based and hash-based approaches on the following criteria: whether the approaches address the three privacy issues identified in Section 3, whether the approaches support semantic-based service matchmaking, whether the approaches can handle the new entities in SBSs easily and the efficiency of the approaches. Table 2 shows the comparison result.

The access-control-based approaches control the appropriate access to the private information to prevent it from being exposed to untrusted service directories. Cryptography-based approaches utilize polymorphic cryptography techniques to enable service matchmaking based on encrypted specifications, such that service directories can not learn about the encrypted private information without correct keys. For the hash-based approaches, there is potential privacy leakage to untrusted service directories. For example, a untrusted service directory can hash keyword "salary" into hashed value, if it receives an advertisement equal to this value, it knows the advertisement is very likely about "salary"; otherwise, it knows the advertisement is not about "salary". In both cases, certain private information of the service provider is leaked. Hence, the hash-based approaches can only partially prevent privacy from being leaked to untrusted service directories.

The access-control-based and cryptography-based approaches can protect the privacy of service providers and service requestors from being divulged to other untrusted parties. The access-control-based approaches will not disclose the private information during service matchmaking unless the other party has the accessing right. Using cryptographic-based approaches, the untrusted parties cannot decrypt the private information without correct keys. However, the hash-based approaches do not address such privacy issues. The private information will be disclosed to other parties who have the same hashed values and know the original specifications, just like the untrusted service directory in the above example. Hence, the hash-based approaches can only partially prevent privacy of service providers and requestors from being leaked to other untrusted parties.

The semantic-based matchmaking algorithms can be performed with the disclosed information using access-control-based approaches. The polymorphic cryptography techniques used by cryptography-based approaches only support limited operations, such as addition and simple comparison, which are not sufficient for semantic-based service matchmaking. Similarly, the hashed descriptions do not preserve the original semantics to support semantic-based service matchmaking using hash-based approaches.

As analyzed in Section 4.1, our SA-AC based approach can easily handle new entities in SBSs with the flexible contextual data based user role assignments, whereas the traditional access-control-based approaches require more expensive policy updating to handle the new entities. The cryptography-based approaches may also require additional key management for new entities. Hash-based approaches can deal with new entities conveniently with the exchanging of appropriate hash functions.

When considering the efficiency of these approaches, our SA-AC based approach and traditional access-control-based approaches both rely on trusted third parties and the enforcement of access control policies requires additional resources. Hence they have lower efficiency than the simpler cryptography-based and hash-based approaches.

The comparison result shows that our SA-AC based privacy-preserving service matchmaking approach distinguishes from other approaches by addressing all the three privacy issues, and supporting both semantic-based service matchmaking and convenient new entity handling.

## 6 Conclusion and future work

In this paper, a situation-aware access control based privacy-preserving service matchmaking approach is presented. Our approach uses SA-AC mechanism to ensure the appropriate disclosure and use of private information in both service advertisements and service discovery requests


IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.

by modeling, specifying, reasoning and enforcing SA-AC policies. It provides an owner-centric mechanism for both service providers and requestors in SBSs to protect their private information during service matchmaking.

To achieve higher accuracy of service matchmaking, we will develop the negotiation mechanisms for participating parties to negotiate over disclosable information, formalize the definition of privacy-preserving strength, and investigate the tradeoff between privacy-preserving strength and matchmaking accuracy. In addition, we will evaluate the overhead caused by the integration of our privacy-preserving service matchmaking mechanism to functionality-based service matchmaking approach presented in [9].

## Acknowledgment

The work reported here was supported by the National Science Foundation under grant number ITR-CYBERTRUST 0430565 and DoD/ONR under the Multidisciplinary Research Program of the University Research Initiative, Contract No N00014-04-1-0723. The authors would like to thank Yin Yin, Wei Gao, Dazhi Huang and Haishan Gong of Arizona State University for many valuable discussions.

## References

[1] Web services architecture. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/, 2004.

[2] Jini architecture specification, v1.2. http://www.sun.com/software/jini/specs/jini1_2.pdf, December, 2001.

[3] J. V. E. Guttman, C. Perkins and M. Day. Service location protocol, v2. http://www.faqs.org/ftp/rfc/pdf/rfc2608.txt.pdf, June, 1999.

[4] Salutation architecture specification (part 1), v2.0c. http://www.salutation.org/spec/Sa20e1a21.pdf, June 1999.

[5] Understanding universal plug and play white paper. http://www.upnp.org/download/UPNP_UnderstandingUPNP.doc, June 2000.

[6] Specification of the bluetooth system. http://grouper.ieee.org/groups/802/15/Bluetooth/core_10_b.pdf, December 1999.

[7] Universal description discovery and integration platform. http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf, September 2000.

[8] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. Semantic matching of web services capabilities. In *Proc. 1st Int'l Semantic Web Conf. on The Semantic Web*, pages 333–347, 2002.

[9] S. S. Yau and J. Liu. Functionality-based service matchmaking for service-oriented architecture. In *Proc. 8th Int'l Symp. Autonomous Decentralized Systems*, pages 147–154, 2007.

[10] S. Pokraev, J. Koolwaaij, and M. Wibbels. Extending uddi with context-aware features based on semantic service descriptions. In *Proc. 1st Int'l Conf. on Web Services*, pages 184–190, 2003.

[11] F. Zhu, M. W. Mutka, and L. M. Ni. Service discovery in pervasive computing environments. *Pervasive Computing*, 4(4):81–90, 2005.

[12] B. Carminati, E. Ferrari, and P. C. K. Hung. Exploring privacy issues in web services discovery agencies. *IEEE Security and Privacy*, 3(5):14–21, 2005.

[13] S. E. Czerwinski, B. Y. Zhao, T. D. Hodes, A. D. Joseph, and R. H. Katz. An architecture for a secure service discovery service. In *Mobile Computing and Networking*, pages 24–35, 1999.

[14] F. Zhu, M. W. Mutka, and L. M. Ni. A private, secure, and user-centric information exposure model for service discovery protocols. *IEEE Transactions on Mobile Computing*, 5(4):418–429, 2006.

[15] L. Kagal, M. Paolucci, N. Srinivasan, G. Denker, T. Finin, and K. Sycara. Authorization and privacy for semantic web services. *IEEE Intelligent Systems*, 19(4):50–56, 2004.

[16] Platform for privacy preferences (p3p) project. http://www.w3.org/P3P/, 2006.

[17] A. C. Squicciarini, E. Bertino, E. Ferrari, and I. Ray. Achieving privacy in trust negotiations with an ontology-based approach. *IEEE Trans. Dependable Secur. Comput.*, 3(1):13, 2006.

[18] J. Kong and X. Hong. Anodr: Anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In *Proc. 4th ACM Int'l Symp. Mobile ad hoc networking and computing*, pages 291–302, 2003.

[19] S. S. Yau, Y. Yao, and V. Banga. Situation-aware access control for service-oriented autonomous decentralized systems. In *Proc. 7th Int'l Symp. on Autonomous Decentralized Systems*, pages 17–24, 2005.

[20] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[21] Web ontology language. http://www.w3.org/TR/owl-ref/, 2004.

[22] S. S. Yau and J. Liu. Hierarchical situation modeling and reasoning for pervasive computing. In *Proc. 4th IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems*, pages 5–10, 2006.

[23] S. S. Yau and J. Liu. Incorporating situation awareness in service specifications. In *Proc. 9th IEEE Inter'l Sym. on Object and Component-Oriented Real-Time Distributed Computing*, pages 287–294, 2006.

[24] Using a first order logic prover with owl. http://wonderweb.man.ac.uk/owl/first-order.shtml, 2004.



IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Arizona State University. Downloaded on July 12,2010 at 20:06:34 UTC from IEEE Xplore. Restrictions apply.