Information Security Technical Report (2005) **10**, 51—58

Information Security Technical Report

Security in the Semantic Web using OWL

Grit Denker [a,*], Lalana Kagal [b], Tim Finin [b]

[a] *SRI International, Computer Science Laboratory, 333 Ravenswood Ave, Menlo Park, CA 94025, United States*
[b] *University of Maryland, Baltimore County, Computer Science and Electrical Engineering, 1000 Hilltop Circle, Baltimore, MD21250, United States*

Abstract Information assurance, security, and privacy have moved from narrow topics of interest to information system designers to become critical issues of fundamental importance to society. This opens up new requirements and opportunities for novel approaches. Meeting this challenge requires to advance the theory and practice of security, privacy, and trust of Web-based applications and to provide declarative policy representation languages and ontologies together with algorithms to reason about policies. This paper summarizes an ontological approach to enhancing the Semantic Web with security.

Introduction

Although today's Internet is a vast information resource, its lack of structure and metadata makes it difficult to extract desired information in a reasonable time. The Semantic Web (Berners-Lee et al., 2001) is a vision of a future Internet in which Web resources are enriched with machine-processable metadata that describes their meaning. This will enable computers to interpret and extract Web content much more effectively and precisely than today's XML-based approaches to allow interoperability. In the Semantic Web (SW), a Web resource's metadata makes it possible to evaluate its appropriateness for a given query, which in turn will lead to greater efficiency of Web resource allocation, despite the daily expansion of Web space.

Berners-Lee (2000) outlines the architecture of the Semantic Web as given in Fig. 1. The layers Unicode, URI, and XML, Namespace, and XML Schema represent current Web technology. RDF and RDF Schema are languages for the description of resources and their types, and thus are basic building blocks for the SW. Above the RDF layer is ontology vocabulary. Ontologies that describe classes or types of things and their relationships form the basis for knowledge representation of

* Tel.: +1 650 859 6058; fax: +1 650 859 2844.
E-mail address: grit.denker@sri.com (G. Denker).

1363-4127/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.istr.2004.11.002

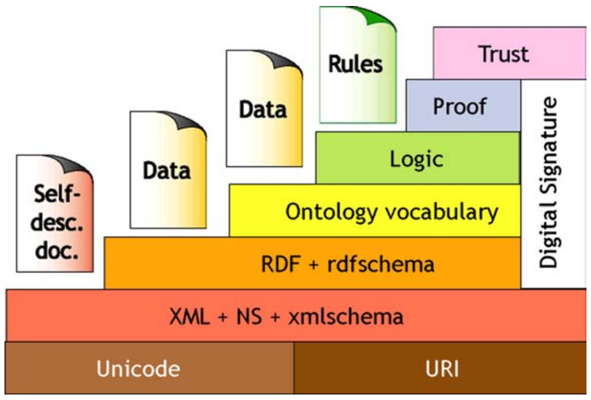

Figure 1 Semantic Web stack.

data. The next layer is logic where new knowledge is generated from existing assertions on the Web by using knowledge representation rules. Finally, the last two layers, and as of yet least developed, are proof and trust. Security issues span all the upper five layers, and digital signatures and encryption are the current state-of-the-art technologies for handling security, although these technologies do not make use of semantic information.

Given the increased importance of the World Wide Web (WWW) for business, industry, finance, education, government and other sectors, security will play a vital role in the success of the Semantic Web. Semantic annotations concerning security aspects of Web resources, Web-based applications, or Web services add value to the Semantic Web when these markups are used in connection with inference systems that support decision processes such as deciding who has access to what resource or selecting services with respect to given security constraints.

The remainder of this paper is organized as follows. In next section we briefly introduce RDF and OWL. Then, on this basis, we illustrate how security and trust can be formalized in OWL and demonstrate reasoning capabilities of a semantic approach to security for the Web. As Web services have gained a lot of momentum in the World Wide Web, Semantic Web services—the equivalent for the Semantic Web—are also becoming more important. Further, we address security and policies for SWSs. We close with a brief summary and point out challenges for future work in the last section.

A (very) brief introduction to RDF and OWL

RDF (Resource Description Framework) (Lassila and Swick, 1999) is a language for representing information about WWW resources. RDF is based on the idea of using Uniform Resource Identifiers (URIs) to identify Web resources in terms of simple properties and property value (Manola and Miller, 2003). URIs can identify things that do not necessarily have network locations or URLs. RDF provides a simple way to make a statement about Web resources based on the RDF triple model. The model contains subject, predicate and object, which correspond to resource, property and value, respectively. The following example taken from Manola and Miller (2003) illustrates the triple model: consider the statement ''http://www.csl.sri.com/users/denker/index.html is created by Grit Denker''. The RDF parts for the statement are: the subject (resource) is the URL, the predicate (property) is the phrase ''created-by'' and the object (value) is the name ''Grit Denker''.

Because of its capability for describing the semantics of data, RDF allows greater resource discovery capability and more meaningful content description.

The Web Ontology Language (OWL) (Web Ontology Working Group; Costello and Jacobs, 2003) is a W3C markup language recommendation for the Web. Because OWL is based on RDF, the purpose of OWL is somewhat similar to RDF. OWL extends RDF schema and is taking a further step in providing a richer vocabulary to express complex relationships and to enhance inference capabilities. Greater degree of inference creates smarter search capabilities, service discovery, and content representation. OWL is an ontology language that can be used to describe the meaning of terminology used in Web documents. OWL supports class definition and subclassing (inheritance), property definition and property hierarchies (including inheritance of domain or range definitions), cardinality restrictions, value restrictions on properties, boolean class expressions (e.g., union, intersection) and many more concepts. OWL is separated into three sublanguages (McGuiness and van Harmelen, 2003)—OWL Lite, OWL DL, and OWL Full—with increasing expressiveness.

Instead of going into detail about OWL language concepts, we will illustrate the use of the language in the context of security ontologies in the following section.

Security in OWL

Web resources and services need to be protected from unauthorized access and software agents accessing online resources on behalf of a user want to be ensured about the privacy of data they disclose to services. Thus, a broad range of security-related notions, such as authentication, authorization, access control, confidentiality, data integrity, and privacy are relevant for SW technology. Currently, low-level encryption, digital signature mechanisms, certification, and public key infrastructures provide a good security infrastructure for Web-based interactions. However, providing higher-level security, especially without prior trust relations in dynamic interactions, relies on a variety of ad hoc mechanisms. This heterogeneity of mechanisms leaves security holes with deleterious effects. The proposed industrial standards on security assume a well-established Web of trust among business-to-business (B2B) partners. For example, there exists a significant body of standardization efforts for security of XML-based Web services, such as WS-security (Atkinson et al., 2002), -trust (Della-Libera et al., 2003), and -policy (Box et al., 2003) at W3C, or SAML of the OASIS Security Services Technical Committee, and the Security Specifications of the Liberty Alliance Project. WS-Security provides a layer of security over SOAP, which is an XML-based protocol for exchanging information primarily used for Web services. WS-Security describes how to attach signature and encryption headers or security tokens to SOAP messages. WS-Policy enables Web services to specify policies in terms of their capabilities, requirements, and other characteristics. WS-Trust describes how existing direct trust relationships may be used as the basis for brokering trust through the creation of security token issuance services.

The standards support low-level security or policy markups that concern formats of credentials or supported character sets for encoding. They do not address semantic user- or application-specific trust tokens and their relations, nor do they allow for expressive policies. The standards deliver to the needs of B2B applications where trusted partners and business relationships have already been established in advance of operation and transactions. However, in a world where more and more public and private services are becoming available online and the vision of cyber-societies is becoming reality, assumptions about pre-established trust relationships do not hold true. The standards are not extensible to more dynamic environments in which simple authentication is not enough, but authentication on user-defined attributes needs to be considered, as ''foreign'' or unknown entities will interoperate with each other across heterogeneous domains and applications using delegation mechanisms. For this, a semantic approach like we take in this paper is a possible solution. Our work is not intended to replace the existing standardization efforts; rather, we propose it as a semantic layer on top of the syntactically oriented standardization.

For privacy issues, the P3P 1.0 Specification (Cranor et al., 2002), a W3C recommendation, is the current standard. P3P is targeted toward controlling how personal information is used on Web sites that one visits. Web sites publish their data-handling practices in XML which can then be matched against client requirements published in APPEL (W3C: APPEL, W3C working draft). Recent work on P3P (W3C: Handling privacy in WSDL 2.0) discusses the extension of P3P toward Web services. For instance, the granularity of the P3P published policy is limited to a URI reference (e.g., an html page). P3P 1.0 is not capable of specifying at the level of individual fields in a form, which has implications for its use in XForms and Web services. However, this extension caters to privacy protection of only the requester of services, whereas privacy is of concern to both the requester and the provider.

While P3P is concerned with privacy protection issues for Web users, EPAL 1.1 (IBM: EPAL 1.1) specifies enterprise-level policies for data objects in the enterprise. EPAL can be thought of as a policy language that specifies rules for enforcing privacy practices promised by a Web site (e.g., through P3P or plain text privacy policy). Enforcement mechanisms for EPAL-specified rules are outside the scope of EPAL.

The above-mentioned frameworks all provide a limited amount of extension to the base data schema that allows addition of domain-specific information (e.g., P3P extension mechanism). We believe that our approach of using policy languages and ontologies allows the inclusion of domain-specific ontologies in a much cleaner way and provides the mechanism for inferring relationships between concepts. Also, such policy languages and ontologies allow privacy policies to be described declaratively over the environment of both the requester and the provider in terms of domain ontologies.

Our approach is to provide a methodical, ontological approach to security markup that proves useful in clarifying which are the high-level security concepts and how they are related to each other. We therefore propose security ontologies in OWL that provide a means for defining security aspects of Web resources (Denker et al., Security for DAML Web services).

One of our high-level security ontologies summarizes various ways in which authentication using credentials can take place. Authentication is often used as a method to decide access requests. Authentication is based on some token, also called *credential,* that the requesting party would know or have. Typical credentials are name-passphrase login, and public and private keys or certificates.

We distinguish between ''SimpleCredential'' and ''ComposedCredential''. The top-level class ''SimpleCredential'' (see Fig. 2) is subclassed to ''Cookie, Login, Key, Certificate, BioMetric'', and ''OneTimePassword'' (subclass relationships are depicted using dotted arrows). All subclasses are pairwise disjoint. ''Public Key'' and ''Symmetric Key'' are disjoint subclasses of the key class. The certificate class is specialized to ''X509Certificate'', and further to the specific class of X509 certificates in the XML Signature (Bartel et al., 2001).

Fig. 2 depicts only classes and their inheritance relationships. Properties and other restrictions are defined in the ontologies as well. For example, the LoginWithPassphrase class has two datatype properties defined as ''loginName'' and ''passphrase'', both of type string (see Fig. 3). We are using the OWL ''restriction'' concept to express that the cardinality on these properties for the login class is constrained to one. That means each LoginWithPassphrase credential has exactly one name and one passphrase.

Abstracting further from credentials, we can specify general security notations as shown in Fig. 4. Several properties are defined for the top class ''SecurityMechanism''. For example, the ontology defines an object property ''syntax'' that has the class ''Syntax'' as range, another property ''relSecNotation'' has the class ''SecurityNotation'' as range, and ''reqCredential'' has the credential class as range. There are various instances for the defined classes. For example, instances of syntax are ''ASCII, OWL, RDF, XML, MIME''; security notations are ''Authentication, Authorization, AccessControl, DataIntegrity, Confidentiality, Privacy, ExposureControl, Anonymity, Negotiation, Policy, KeyDistribution''; and ''X.509'' is an instance of the KeyFormat class.

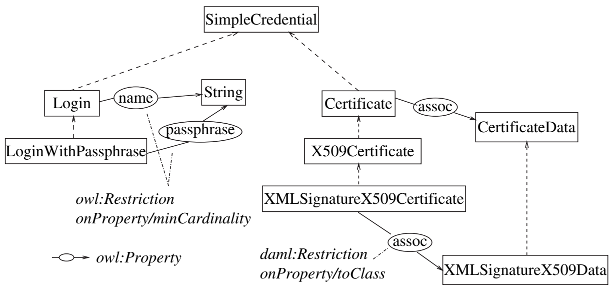

Figure 3 A credential ontology (some properties).

''XML-DSIG'' is an instance of the signature class and ''OpenPGP-Enc'' is of type ''Encryption''. Specific protocols such as ''X-KRSS, X-KISS, Kerberos'', or ''SOAP'' are defined as instances of the appropriate protocol subclasses that satisfy certain restrictions.

The security ontology summarizes at a very high abstraction level many of the commonly used security-related notations. Such notations can be used to describe user, agent, or service security policies.

The ontologies are not only useful for description of Web resources, but they also provide a basis for reasoning. For example, we used the concept of OWL restrictions to define certain security characteristics. A restriction class in OWL constrains the range of an object property. For example, a restriction on the property ''relSecNotation'' yields an ''AuthenticationSubClass'', that is, a class that has as one of its related security notations ''Authentication''. Thus, if a Web resource has the authentication class as one of its requirement, the resource requires the user to authenticate. This information can be used in reasoning about whether a user's preferences match the resource's requirements. Consider, for example, a user who is interested in finding information about his hobby, but is not willing to sign up to yet another online service. This user would not be interested in a Web service that requires all users to sign in for authentication

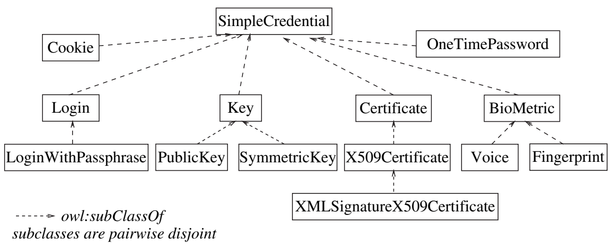

Figure 2 A credential ontology.

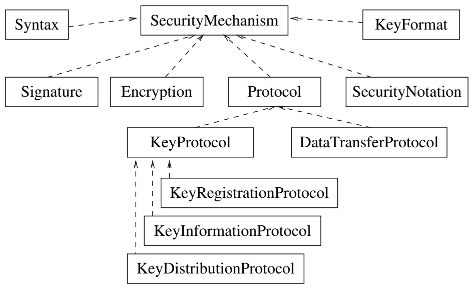

Figure 4 A security ontology.

purposes. A software agent for that user, equipped with the policies of ''not using services that require authentication'', could upon finding a Web service, read its metadata and, employing Semantic Web reasoning techniques, decides whether the service adheres to the user's policies.

Security for Semantic Web services

An increasing number of organizations are endorsing Web service (WS) technology to extend corporate resources to customers and partners and to leverage resources of others. Although Web services, based on XML technology, allow interoperability, they do not support efficient and flexible search, allocation, composition, runtime monitoring, or invocation of those Web services. Aspects of security, availability, and trustworthiness of Web services are not adequately addressed in a sufficiently abstract level by current industry standards. Through the exploration of how semantically rich annotations will facilitate these tasks, Semantic Web services promise to provide solutions. Semantic Web services (SWS) is a new research direction to support means by which services can be given richer semantic specifications. Richer semantics can provide fuller, more flexible automation of service provision, and use and support the construction of more powerful tools and methodologies.

As Web services connect independently controlled domains, across application and organizational boundaries, issues related to service protection and security, reliability of results, or validity of source and cost become important. One crucial issue is the *dynamic nature* of many transactions where service requesters and service providers interact *without any prior direct trust relationships*. A prominent example is e-science and Grid services where data, computer hardware and instruments, and computational processes are brought together in a dynamic fashion to achieve the best results. In these situations, trust relationships must be established on the fly and for a limited purpose and time. To facilitate such flexible establishment of trust, there is a need for formalisms to express security policies in ways that software agents can access them and use them to make choices as to which service to engage.

OWL-S

Our work is developed in the context of Web service descriptions in OWL-S (OWL-S coalition: OWL services). OWL-S and other related work may be viewed as efforts to lay the foundations for the most effective evolution of Web-service-related capabilities that can be supported with current and maturing technologies. At the same time, our goal is to promote the rapid adoption of semantically expressive technologies that are already well understood, and there is much that can be done in the near term. Therefore, mechanisms are available by which OWL-S can be used along with the dominant Web services standards, such as WSDL.

OWL-S is a set of ontologies for describing capabilities, interfaces, and other details of Web services and has been designed to facilitate automated Web service discovery, composition, and invocation. An OWL-S description comprises a profile that provides a general description of the Web service, a process model that describes how the Web service performs its tasks and the Web service interaction protocol, and a grounding that specifies how the atomic processes in the Process Model map onto WSDL (WSDL:TR) representations (see Fig. 5).

There is no explicit place for security annotations and policies in OWL-S, but the natural extension toward security is to link policies to the profile. The rationale for this is that policies specify general properties of the Web service rather than properties that are specific to any process. In collaboration with researchers from Carnegie Mellon University (Katia Sycara and Massimo Paolucci) and the University of Maryland, Baltimore (Tim Finin and Lalana Kagal), we have begun addressing security issues for SWS by providing *semantically meaningful*, declarative, machine-processable *policy descriptions and associated algorithms and tools* to match policies of service requesters with policies of service providers. We make use of semantically rich annotations as provided by Semantic Web services to achieve this goal. In Denker et al., Security for DAML Web services, we proposed two properties for SWS, namely ''securityRequirement'' and ''securityCapability''. The range of these properties is the ''SecurityMechanism'' class from the ontologies presented in the previous section. These

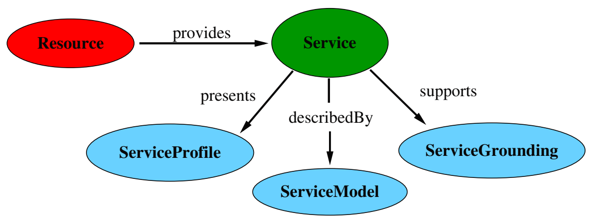

Figure 5 Top level OWL-S ontology.

properties allowed us to describe WS and agents in terms of requirements, such as ''service requires authentication with X509 certificate'', and their capabilities, such as ''Agent can provide smart card credential''. These descriptions are static in the sense that they are not described in terms of actions, but rather in terms of security classes (such as the restriction classes described in the previous section). To gain more expressiveness for security characterizations, we proposed to extend the ontological approach with policies.

Policies for SWS

In addition to the abstract security notations that we illustrated, an aspect that is gaining increasing interest in the Semantic Web is generic security policies. Consider, for example, privacy issues in online transactions. Today, online transactions have humans in the loop, where individuals peruse the various policies on privacy concerns and contracting assumptions that online sites publish, and then choose whether or not to make use of the service. For automated use of Web resource, e.g., via software agents acting on behalf of users, it is therefore important to provide technology that allows software agents to express their preferred policy with respect to their own privacy concerns (i.e., the privacy concerns of the user on whose behalf they are acting) and to validate that the expressed policy would be carried out by the service providers.

In Kagal et al. (2004) we propose to extend the security framework introduced in the previous section through the addition of annotations about the security and privacy policies of services expressed in Rei which allows rules and constraints to be defined over domain-specific ontologies. This policy information is used in service selection and invocation. We propose policies as an extension of the security requirements of services and suggest the addition of a property, called *policyEnforced,* defined as a subproperty of *securityRequirement* (see http://www.daml.org/service/owl-s/security.html, serviceSecurity.owl). The policyEnforced property describes the different policies that must to be enforced for the correct execution of service.

The key idea is to integrate policies into the representation of Semantic Web services to provide security at various levels. Policies provide specifications that restrict who can use a service under what conditions, how information should be provided to the service, and how provided information will be used later. These specifications include constraints over different attributes of the requester, the service, and the content in general. For example, the policy of a stock quote service is that the service is available only when the stock market is open. This policy describes the conditions under which the service is available, and these conditions are contextual as they include the time when the service is invoked. Along with restricting access to services, policies can be used to safeguard a client's or a service's privacy and confidentiality.

We address three kinds of policy: authorization, privacy, and confidentiality. An *authorization policy* is defined as a set of rules that restrict access to a service. These policies constrain access to the service to only certain clients and under certain circumstances. For example, an authorization policy of a university printer service states that only members of the CS department are allowed to access the service. In this example, the policy describes the credentials required for access, which is the membership of the CS department.

Privacy policies specify what information can be exchanged, the legitimate uses of the information, and the conditions under which this exchange is possible. For instance, the privacy policy of a user states that the user's Social Security number should not be disclosed to anyone. Privacy policies specify restrictions not only on data exchanged, but also on the recipient after the receipt of data. Consider, for example, a service that states that it will not distribute any details it receives as input. This might be an important requirement for a requester who requires privacy. This privacy policy can be interpreted as an obligation on the Web service and can then act as a contract. In this way, if after invocation, the service does provide the user's details to a telemarketer, the user can take legal action against the service based on the policy.

Closely linked to privacy policies are *confidentiality policies* that describe cryptographic characteristics of the input and output parameters of a service. A requester's confidentiality policy might state that all communication must be encrypted. This policy is matched against the process model of the service to verify that the service indeed performs as required.

Policies are basically rules of behavior over different attributes of the requester, service, and any context. To describe policies, we use Rei (Kagal et al., 2002, 2003), a logic-based language for policy specifications based on RDF and RDF Schema (RDFS). Rei is a declarative policy language that includes notions of logic-like variables for describing different kinds of conditions that are not directly possible in OWL. For example, it is possible to specify in Rei that the uncle of the owner of the pay-for-use service is authorized to use the service free of charge. These variables are defined as type ''Variable'' of the Rei ontology, and the unification of variables is carried out in the same fashion as Prolog. Rei is modeled on the basis of deontic concepts of permissions, prohibitions, obligations, and dispensations. These constructs have three main attributes: actor, action, and constraint. Constraint specifies conditions over the actor, action, and any other context that must be true at the tune of the invocation of the action or service. These deontic constructs are linked to policies through the ''granting'' class that allows additional constraints to be attached to the deontic rule. These constructs allow Rei to model authorization, privacy, and confidentiality policies in terms of what an entity can/cannot do or must/must not do.

Associated with the policy language is a policy engine that interprets and reasons over the policies and domain information to make decisions about applicable negative/positive permissions and prohibitions. So far, we have concentrated on the following two tasks: (i) provision of authorization, privacy, and confidentiality policies for SWSs, and (ii) design of matchmaking and compliance checking algorithms on the basis of semantic annotations, policies, and properties of the requester, the service, and the general context.

Other work in the area of trust and privacy policies for the Semantic Web such as Gandon and Sadeh and Bradshaw et al. (2003) is also relevant for our work, although this work is not specifically targeted toward Semantic Web services.

There exist other policy languages, such as KAoS (Bradshaw et al., 2003), Ponder (Damianou et al., 2001), Security Policy Language (SPL) (Ribeiro et al., 2001) and Delegation Logic (DL) (Li et al., 2000). KAoS allows policy specification on the basis of OWL constructs. The advantage of our approach is the explicit use of variables that make policy expressions more expressive. Ponder is a flexible and expressive policy specification language, but it does not lend itself to being useful for Semantic Web services as it is more of a syntactic language. Ponder allows definition of positive and negative authorization policies (access control), information filtering (transforming requested information into a suitable format), and delegation positive and negative. SPL is an access control language for security policies with complex constraints. DL is a logic-based, computationally tractable, knowledge representation that deals with authorization in large-scale, open, distributed systems that has been widely deployed.

The policy languages described above have some common limitations that make them difficult to extend for the environment in which semantic Web services will be used. (1) They do not handle delegation as required in dynamic environments. The policy languages we propose add a series of constraints at every level of delegation. This allows control over not only who can delegate but also to whom they can delegate and under what conditions. (2) They do not support justification and advising that is required for greater autonomy. (3) They use syntactic languages that cannot be easily modified to work with different domain-specific information.

Conclusions and future work

Information assurance, security, and privacy have moved from narrow topics of interest to information system designers to become critical issues of fundamental importance to society. As such, they also play an important rule in Web-based applications and newer technology such as Semantic Web applications. These applications need the capability for agents, devices, and services to seamlessly interact while preserving appropriate security, privacy, and trust.

Meeting this challenge requires realizing four high-level objectives: (1) to advance the theory and practice of security, privacy, and trust of Web-based interactions by providing technology for trust in the Semantic Web and trust worthy, semantically annotated Web services; (2) to provide declarative policy representation languages, ontologies, and inference algorithms for security, trust and privacy management, enforcement, and negotiation; and (3) to prototype software tools allowing system designers and end users to both specify and verify policies for trust and privacy.

In this paper we have mainly addressed the knowledge representation and some of the reasoning issue for trust and security in the Semantic Web. Enforcement mechanisms, automatic negotiation and tools for SW and SWS security are much less developed and provide ample opportunities for future research. These opportunities include: first, building semantic abstractions on current and emerging Web standards for security, trust, and privacy to ensure wide societal adoption and deployment; second, incorporating policy models into the descriptions of services and agents that act on behalf of users enables policy-based service discovery, selection, composition, and monitoring;

third, designing algorithms that use declarative policy descriptions enables reasoning about policy compatibility and policy negotiation, and facilitates implementing enforcement mechanisms transparent to end users. Finally, users, system designers, and administrators can use intuitive policy engineering tools to specify and validate effective policies for security, privacy, and trust.

The technical advances achieved will significantly impact our ability to engineer open and flexible information services that are secure, support enforceable privacy policies, and incorporate models of trust. This satisfies growing requirements in education, commerce, and government applications.

References

Atkinson B, Della-Libera G, Hada S, Hondo M, Hallam-Baker P, Klein J, et al. Web services security (WS-Security). <http://www-106.ibm.com/developerworks/webservices/library/ws-secure/>; 2002.

Bartel M, Boyer J, Fox B, LaMacchia B, Simon E. XML-signature syntax and processing rules. <http://www.w3.org/TR/2001/PR-xmldsig-core-20010820/>; 2001.

Berners-Lee T. Semantic Web—XML2000. <http://www.w3.org/2000/Talks/1206-xml2k-tbl/slidel0-0.html/>; 2000.

Berners-Lee T, Hendler J, Lassila O. The Semantic Web. Scientific American; 2001

Box D, Curbera F, Hondo M, Kale C, Langworthy D, Nadalin A, et al. Web services policy framework (WS-Policy). <http://www-106.ibm.com/developerworks/library/ws-polfram/>; 2003.

Bradshaw J, Uszok A, Jeffers R, Suri N, Hayes P, Burstein M, et al. Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads. In: AAMAS'03, July 14—18 Melbourne, Australia.

OWL-S Coalition: OWL services. <http://www.daml.org/services/>.

Costello R, Jacobs D. A quick introduction to OWL Web Ontology Language. <http://www.xfront.com>; 2003.

Cranor L, Langheinrich M, Marchiori M, Presler-Marshall M, Reagle J. Platform for privacy preferences (P3P); 2002.

Damianou N, Dulay N, Lupu E, Sloman M. The Ponder policy specification language. In: Lecture Notes in Computer Science, 1995, 2001.

Della-Libera G, Dixon B, Garg P, Hallam-Baker P, Hondo M, Kaler C, et al. Web services trust language (WS-Trust). <http://www.106.ibm.com/developerworks/library/ws-trust/> 2003.

Denker G, Kagal L, Finin T, Paolucci M, Sycara K. Security for DAML Web services: annotation and matchmaking. In: Fensel D, Sycara K, Mylopoulos J, editors. Proceedings of the 2nd international Semantic Web conference (ISWC 2003) Sanibel Island, FL, LNCS 2870. Springer; October 2003.

Gandon F, Sadeh N. A semantic e-wallet to reconcile privacy and context awareness. In: Fensel D, Sycara K, Mylopoulos J, editors. Proceedings of the 2nd international Semantic Web conference (ISWC 2003) Sanibel Island, FL, LNCS 2870. Springer; October 2003. p. 385—401.

<http://www.w3.org/TR/wsdl/>.

IBM. EPAL 1.1. <http://www.zurich.ibm.com/security/enterprise-privacy/epal/Specification/index.html>.

Kagal L, Finin T, Joshi A. A policy language for a pervasive computing environment. In: IEEE 4th international workshop on policies for distributed systems and networks; 2002.

Kagal L, Finin T, Joshi A. A policy language for pervasive systems. In: Fourth IEEE international workshop on policies for distributed systems and networks; 2003.

Kagal L, Finin T, Paolucci M, Srinivasan N, Sycara K, Denker G. Authorization and privacy for Semantic Web services. In: IEEE Intelligent Systems; 2004.

Lassila O, Swick R. Resource Description Framework (RDF) model and syntax specification. <http://www.w3.org/tr/rec-rdf-syntax/>; 1999.

Li N, Grosof B, Feigenbaum J. A practically implementable and tractable delegation logic. In: Proceedings of 2000 IEEE symposium on security and privacy (S&P'OO). IEEE Computer Society; 2000. p. 27—42.

Manola F, Miller E. RDF primer. <http://www.w3.org/TR/rdf-primer/>; 2003.

McGuiness DL, van Harmelen F. OWL Web Ontology Language overview. <http://www.w3.org/TR/owl-features/>; 2003.

Ribeiro CN, Zuquete A, Ferreira P, Guedes P. SPL: an access control language for security policies with complex constraints. In: Network and Distributed System Security Symposium (NDSS'Ol); 2001.

Web Ontology Working Group: OWL. <http://www.w3.org/2001/sw/WebOnt/>.

WSDL: Technical report, <http://www.w3.org/TR/wsdl/>.

W3C: APPEL. W3C working draft. <http://www.w3.org/TR/P3P-preferences/>.

W3C. Handling privacy in WSDL 2.0. <http://www.w3.org/TeamSubmission/2004/SUBM-p3p-wsdl-20040213/>.