

Towards Secure Collaboration on the Semantic Web

Joon S. Park
School of Information Studies
Syracuse University
Syracuse, NY 13244-4100
Tel. 1-315-443-6887
jpark@syr.edu

ABSTRACT

Web technologies enable collaborative work to be done more efficiently and effectively. A user can share resources with others on the Web and perform his or her job based on a pre-defined policy for collaboration. During the collaboration, the user may need to create new resources, merge, split, exchange, or update resources created by other users. To support these services on the Web, we need machine-understandable as well as machine-readable metadata about the resources. The concept of a Semantic Web has been introduced to satisfy this requirement. Although the Semantic Web will provide more accurate and efficient services on the Web, it also introduces new problems that were not considered before, especially, in regards to security, interoperability, and transparency to users and organizations. In this paper, we discuss the requirements to support secure collaboration on the Semantic Web. We mainly focus on identification and analysis of the security problems associated with the Semantic Web, while suggesting possible solutions to each problem.

1. INTRODUCTION

The Internet has created a virtual common workplace that facilitates collaboration; however, most data on the Web is simply machine-readable, which is not sufficient to automate processing of Web resources, rather than machine-understandable. To provide automatic and accurate Web processing in reliable manners, the data must be made machine-understandable. For this purpose, the concept of the Semantic Web [BHL01, SWeb, SWO], which would provide machine-understandable metadata about resources, has been introduced. The Semantic Web provides the abstract representation of data on the WWW, based on the RDF (Resource Description Framework (RDF, [LAS98, LS99]) standard and other standards to be defined.

Advances in the use of metadata achieved through the W3C Metadata Activity [WMA], now the W3C Semantic Web Activity [SWeb], have created strategies for describing and classifying resources such as contents, technology, and services. For example, the Platform for Internet Content Selection (PICS [PICS]) is a suite of specifications that enables people to assign various computer-readable “labels” to digital material. Individual users or machines can utilize these labels either to filter out or to pass materials that match pre-set criteria. While originally intended as a means for parents, schools, and libraries to decide which materials would be acceptable at their sites, PICS eventually also led to RDF (Resource Description Framework).

RDF was developed by several W3C member companies to describe resources in machine-understandable metadata, using XML as an interchange syntax to support the interoperability of metadata of Web resources. The XML syntax is one of many possible syntax representations for RDF. New languages to represent the same RDF may emerge (e.g., Notation 3 [Notation3]). RDF is an application-neutral mechanism for describing information about any application. It neither makes assumptions nor creates semantics for a particular application domain. The resources are described in XML in terms of a collection of properties where each property has its type and value. However, the resources themselves can be of any type, including XML and non-XML. RDF enables automated processing of Web resources through machine-understandable semantics for metadata. It provides the basic building blocks for the Semantic Web. The metadata via RDF on the Semantic Web enables more accurate and efficient services, such as resource recovery, content cataloging, knowledge sharing, protecting intellectual property rights, and enforcing policies for a particular Web site.

To provide automatic and accurate content services on the Web, content descriptions, which can be read and understood by machines, is indispensable. If the metadata provides sufficient information in a machine-understandable format, computers will be able to make a decision and proceed with their jobs automatically and accurately, based on the metadata. The metadata and the actual contents can be described within the same entity (e.g., an XML file) and stored in the same place, such as a Web server. Alternatively, they can be described in separate entities and stored in different places. For instance, a document is generated in a PDF file stored in a Web server, while its mobile policy is described in an XML file and stored in a policy server. The former approach (with single entity) supports higher mobility, while the latter (with separate entities) is better in terms of maintenance (e.g., policy change), but it requires a linking mechanism between the contents and the corresponding metadata.

Using metadata allows information about Web resources to be represented, but also introduces new issues and requirements for the new service to be available to users. In this paper, we identify the new requirements for supporting secure collaboration on the Semantic Web. We mainly focus on the security problem analysis, providing a few suggestions on how to address these problems. In summary, to support secure collaborative work on the Semantic Web, we need to consider the following issues, which are discussed in detail in the following subsections in turn.

- Interoperability of metadata
- Who adds the metadata concerning contents?
- Protection and maintenance mechanisms for metadata as well as contents
- Policy enforcement

- Efficient access control for collaborative work

In addition to these issues, the Semantic Web shows common concerns, such as user and server authentication, intellectual property rights, denial of service, monitoring, and communication protection. While these issues are of importance, they are not specified to the Semantic Web. We focus on newly introduced requirements with the use of the Semantic Web to support secure automatic processing for collaborative work. Some of those issues or even possible solutions have already been addressed by other researchers directly or indirectly [BF01, HM01, PH01]. We discuss a comprehensive set of those new requirements with descriptions in this paper. While these issues are discussed in the context of a medical collaboration in this paper, we believe that our approaches can be applied to any Semantic Web applications. The ideas that we introduce in this paper have been applied to real systems, where users dynamically join or leave a service domain for collaboration, such as inter-organizational workflow [PKF01] or peer-to-peer computing environments [PH03a, PH03b].

2. INTEROPERABILITY OF METADATA

A collaborative work can be performed by not only among participating individuals in an organization, but also by collaboration among different organizations. In such cases, organizations may not necessarily use the same metadata format to describe the same set of resources. While a metadata-provider creates metadata for resources in Organization A, we can assume that the metadata created in Organization B is created by someone or something else, and does not necessarily have the same syntax and format as that of the first metadata. For a simple example, the metadata for patient Alice's information in Hospital A may look like:

```
<PatientInfo rdf:about="Alice">
  <dc:Hospital>Hospital A</dc:Hospital>
  <dc:DateOfBirth>12/31/2002</dc:DateOfBirth>
  <dc:Sex>Female</dc:Sex>
  <p:PrimaryDoctor>
    <p:name>Chris</p:name>
    <p:specialty>Pediatric</p:specialty>
  </p:PrimaryDoctor>
  <dc:MedicalHistory>
    :
    :
  </dc:MedicalHistory>
</PatientInfo>
```

On the other hand, the metadata for patient Bob's information in Hospital B may look like:

```

<PatientInfo rdf:about="Bob">
  <dc:Hospital>Hospital B</dc:Hospital>
  <dc:DateOfBirth>
    <dc:Date>01</dc:Date>
    <dc:Month>January</dc:Month>
    <dc:Year>1950</dc:Year>
  </dc:DateOfBirth>
  <dc:Sex>Male</dc:Sex>
  <p:PrimaryPhysician>
    <p:name>Dave</p:name>
    <p:specialty>Internal Medicine</p:specialty>
  </p:PrimaryPhysician>
  <dc:MedicalHistory>
    :
    :
  </dc:MedicalHistory>
</PatientInfo>

```

The difference in the terminologies is that the former one uses *PrimaryDoctor* to represent the primary medical expert for the patient while the latter uses *PrimaryPhysician* to represent the same job. Furthermore, the first metadata states the patient's birthday in one line, while the second metadata states the patient's birthday in three separate lines. The differences in the terminologies and formats can happen to the same patient, whose records are in multiple hospitals. Therefore, we need a way to help machines understand that both *PrimaryDoctor* from Hospital A's metadata and *PrimaryPhysician* from Hospital B's metadata refer to the same role – the person who gives medical treatment, and that even though the formats are different, they both pertain to the date of birth of the patients. Many researchers have been working to satisfy this requirement through the use of ontologies, providing a common way to describe resources and their relationships to others [PH01]. Ontology will play a major role in supporting the Semantic Web, which provides machine-understandable semantics of data.

For any Semantic Web application to be useful and effective, domain-specific ontologies need to be developed. For instance, in order for our example to work properly among different organizations, we need to develop an ontology for medical terminologies and data formats. To address the problems of security policy, we could develop another ontology to describe security related properties. While a medical ontology would only be useful in specific applications, a security ontology would have broad use in any application that had security concerns. For instance, if an application-dependent security policy is mapped to a set of generic operations (e.g., read, write, execute, etc.) thorough the ontology, the generic mechanisms can enforce the application's policy automatically (described in Section 5). Since current XML specifications for this purpose [DAML, DSC01, HM01, LAS98, LS99] have limited

capabilities, we need to extend the existing specifications with a rich set of expression for creating ontologies and enabling the instantiation of those ontologies by means of metadata.

3. METADATA PROVIDER

To provide automatic and accurate content services on the Web, resource descriptions, which can be read and understood by machines, is indispensable. Who then provides the metadata? The current version of the Semantic Web does not specify how the metadata will be created, and most importantly, who will create the metadata used for making information machine-understandable as well as machine-readable.

Suppose patient Bob visits a doctor's office. If it is his first visit to the office, he is required to fill out a paper registration form, including the patient's name, address, telephone number, doctor's name he wants to see, health insurance information, and so on. Then, secretary Harry enters the registration information that patient Bob just wrote in the paper form into their database system. In the meantime, nurse George may measure patient Bob's weight, height, and blood pressure, and write those numbers in another paper form. Finally, doctor Chris examines Bob and writes some medical notes, describing Bob's symptoms, prescriptions, and medical treatments, in another paper form. Later, practitioner Eve enters doctor Chris's and nurse George's notes into their database system, where patient Bob is registered.¹

The above example shows that many people are involved to manage the patient's medical record and other related information. It could be assumed that the creator of the resources (patients, doctors, and nurses in our example) will be responsible for creating the metadata tags needed for agents and other machines to understand the information pertaining to these resources. However, we cannot simply assume that the resource creators know how to add the metadata, particularly in XML based on RDF, to the resources they create. Most users do not know (and probably do not want to know) how to provide metadata about the resources. However, they do want access to easy-to-use, secure services. Although some advanced wireless technologies, such as wireless PDAs, could help doctors and nurses to input patients' records directly into their database systems, we would still need persons who maintain corresponding metadata for each record.

Therefore, we need a third party who creates and maintains the metadata and a mechanism to support the trust in the third party. There can be multiple authorities for the metadata about the same resources (This issue will be discussed in detail in section 5). Similar to the way some applications generate tags for HTML, we can envision tools that automatically generate metadata for our resources based on specific, pre-defined rules and the needs of the individual or organization and application.

When using a third party to provide the metadata, content owners need to form a trust relationship with the metadata provider. The trust relationship implies that the content owner trusts the identity of the

¹ In real life, some doctor's offices and even hospitals may not use database systems to keep their patients' records yet.

content provider and his ability to provide accurate metadata for their content, which also requires the content owner to trust that the metadata provider will not perform actions he shouldn't. For instance, the content owner trusts that the metadata provider will not change or disclose the content in unauthorized ways. The concept of trust of identity also implies that only authorized persons will be able to create metadata for the content. Therefore, the trust mechanism should include a means to establish identity, enforce access control, and provide integrity and confidentiality services.

The metadata and the actual contents can be described within the same entity (e.g., an XML file) and stored in the same place, such as the peer's machine. Alternatively, they can be described in separate entities and stored in different places. For instance, a document is generated in a PDF file stored in one machine, while its mobile policy is described in an XML file and stored in another machine. The former approach (with single entity) supports higher mobility, while the latter (with separate entities) is better in terms of maintenance (e.g., policy change), but it requires a linking mechanism between the contents and the corresponding metadata. If both the contents and metadata are described in XML, we can provide fine-grained access control [KFP01] to each field in the XML files.

4. PROTECTION AND MAINTENANCE

Technically, both the actual resources (e.g., medical information) and their corresponding metadata (e.g., support information) can be contained in the same file, such as an XML file. Alternatively, the resources and metadata can be distributed in separate files, providing links between them via URIs. In either case, for secure processing on the Semantic Web, it is imperative to protect the metadata as well as resources from possible threats, providing integrity, confidentiality, access control, and other security services. For instance, in the above example, if the required condition, such as roles, for access to patient Alice's record described in a metadata, is changed in an unauthorized manner, the resource (Alice's medical record in our example) can be accessed by unauthorized users.

Some protection mechanisms for the resources can be provided by the resource creators, using existing cryptographic technologies, such as encryption or digital signatures. For instance, doctor Chris digitally signs the prescription he writes for patient Alice and pharmacist Dave verifies Chris's signature on the prescription using Chris's public key to check if the prescription has not been modified in an unauthorized manner. Similar techniques can be used to protect the corresponding metadata. Hence, to protect the metadata itself, the metadata creator, Harry, can encrypt or digitally sign the metadata he created. It is simple when all the metadata is controlled by a single authority. Then, Harry can simply sign the complete metadata in the XML file. The simplest case would be when all the fields in both resource and metadata could be signed by one single authority. In this case, the protection and maintenance mechanisms are monolithic.

An XML file can be encrypted or digitally signed as a whole and sent securely to one or more recipients. We can simply use conventional cryptographic technologies for the entire file. However, problem arises when parts of an XML file need to be signed or encrypted, probably by different users. Typically, however, the resource creators are not metadata creators as we described in section 3. This case is similar to the multilevel information structure in a military message, which contains multiple single-

level units, each with its own classification [LHM84]. To support this service, a trusted tool is required for the users to deal with the multilevel information. Likewise, different XML files or even different portions in the same file may require different levels of protection and separate controls. Some portions must be strongly protected. For instance, if Harry maintains the metadata for patient Alice's registration information (e.g. name, address, date of birth, etc.) while another metadata provider, Ivy, maintains the metadata for patient Alice's medical information (e.g. medical history, prescriptions, symptoms, etc.), neither Harry nor Ivy can simply sign all the metadata related to the even same patient, Alice. Furthermore, someone (say, John) should maintain the metadata about other metadata (e.g. doctor Eve's comments on doctor Chris's prescription for patient Alice). Obviously, the monolithic protection and maintenance mechanisms are not a good solution in this case.

Therefore, we need new mechanisms to support multiple authorities for the same XML file, providing the links among them and autonomous protection and maintenance mechanisms [PS00]. Currently, many XML specifications are available or under development. Some [AXML, S2ML, HM02, XKMS, XrML] of those specifications define how to exchange security-related data, such as access control, authentication, or authorization information, among system components in XML format, while some [XMLE, XMLS] define how to protect XML file itself via digital signatures or encryption. However, it has not been clearly defined how to provide secure Semantic Web services, especially, when different authorities are involved in the same XML file.

5. POLICY ENFORCEMENT

Various individuals have different types of access to the resources depending on the job of the individual. These users' access to the collaborative work should be allowed based on pre-defined policies and privileges. Some systems should enforce some application-specific policies that are absent in other applications [LHM84]. Policies can be defined in a dedicated place, such as a policy server, and referenced by system components, such as Web servers, to make correct decisions [BF01]. Alternatively, policies can be distributed to the places where they are needed. This decreases the vulnerability to single-point-failure and provides better performance. The ultimate way to distribute the policies is bundling the policies with the resources to which the policies apply in the same file. These mobile policies move along with the resources so that any system components (e.g. Web servers) can enforce these policies automatically without asking other components. How then can we define these policies? There can be different ways to describe the policies. To enforce policies defined in a particular language, system components need to interpret the policies into proper implementation logic, which is not always possible because of language incompatibility. If we define the policies in a universal language, we can solve this problem. We believe that a standard metadata format can be used as a framework for this purpose, probably with some extension for a richer expression. In summary, if resources travel with mobile policies, which are defined in a standard metadata format, machines can understand the policies and enforce them automatically on the Web.

A similar approach can be applied to users' privileges. To complete their tasks for the collaborative work, users need some privileges over the resources. In classical approaches, we have considered four different privileges: Read, Write, Execute, and Delegate. Read and Write privileges give the user the ability to observe and modify the corresponding resources. Execute privilege, which applies only to

executable content, gives the user the ability of triggering some actions of the resources. Finally, a user with Delegate privilege on a resource can pass her privilege (e.g. read) on the resource to other users. To properly delegate privileges, the user must have Delegate privilege and the privilege she is attempting to give to other users. For example, if doctor Chris wants to delegate his Read privilege of patient Alice's medical history to his nurse, he must have both Delegate and Read privilege to the patient Alice's medical record. These classical privileges can also be joined into an abstract privilege so they can be assigned to the users as a single unit. The nature of an abstract privilege depends on applications. For instance, it is reasonable to support doctor, nurse, and patient privileges in our medical example, while credit, debit, and balance transfer privileges are supported in an accounting application. These privileges, bundled together as abstract privileges can be assigned to or revoked from the users based on their context in the collaborative work. This is quite beneficial to system administrators and project managers. Therefore, if we define abstract privileges, which are application-specific, in a universal language in terms of generic operations, machines can understand how to deal with the abstract privileges for their automatic processing.

6. EFFICIENT ACCESS CONTROL FOR COLLABORATIVE WORK

If a system supports many users, we should consider more efficient and secure access control mechanisms for both inside and outside users. This becomes more serious when multiple organizations are involved in the same collaborative project. Some of the users may have common functionality in one project while each user may be involved in multiple projects. A user's privileges need to be changed based on his or her context for the project. Each project may have static or dynamic constraints. For instance, nurse George is allowed to read patient Alice's medical record, while she is not allowed to read patient Bob's medical record. Furthermore, doctors and nurses may have different privileges in different hospitals. There can be more complex cases for providing minimum privileges to the right users at the right time.

We need an efficient mechanism to control "Who has which privileges for which resources under what conditions?" Technically, we could use the conventional identity-based access control mechanism for this purpose. This could work for a small project, where a small number of users (perhaps in the same organization) are involved and job assignment is stable. However, for a large system, which supports many users from different organizations, the identity-based access control mechanism is inefficient and too complicated to manage, because the direct mapping between users and privileges is transitory. Furthermore, each organization does not know the identities of the users from other organizations who are involved in the collaborative project. The organization could share their user databases and recognize the users from the other organizations. This is not a sound solution because it applies to all the users while usually, only some of the users in each organization are involved in the collaborative project. Alternatively, each organization could setup a database for only participating users and share it with others. However, since the assignment between users and their job functionalities for the project are changed frequently by each organization, updating and synchronizing each database with others is not always easy.

Therefore, we should separate the mappings between users and privileges through common job functionalities, such as roles [PAS01, SCFY96]. Usually, functionality-privilege mapping is more stable

(of course, it can be changed when necessary) than user-functionality mapping, because job responsibilities for a collaborative project do not change frequently while users' job functions change quite often. The system makes access control decisions based on the users' job functions instead of their identities. This provides an efficient access control mechanism to the system and resolves the scalability problem. To support role-based access control among different organizations, we need to provide a common access control ontology for the collaborative project. Furthermore, we will provide a fine-grained access control [KFP01] mechanism to the content services. For instance, different fields in the same XML file may require different roles to be accessed.

7. CONCLUSIONS

In this paper, we have identified and discussed several new requirements necessary to support secure automatic processing on the Semantic Web. The issues examined in the paper are: interoperability of metadata, providing metadata, metadata protection and maintenance, automatic policy process, and efficient access control for collaborative work. For each requirement we have looked at several possible ways of addressing the issues. While we have used an example for collaboration in hospitals to describe our approaches, the issues discussed here are also of concern to other applications of the Semantic Web for collaborative work. The ideas that we introduced in this paper have been applied to real systems, where users dynamically join or leave a service domain for collaboration, such as inter-organizational workflow [PKF01] or peer-to-peer computing environments [PH03a, PH03b].

8. REFERENCES

[AXML] *AuthXML*, <http://www.authxml.org/>.

[BHL01] T. Berners-Lee, J. Hendler, and O. Lassila. *The Semantic Web*. Scientific American, 284(5):34-43, 2001.

[BF01] E. Bertino, S. Castano, and E. Ferrari. *Securing XML Documents with Author-X*. IEEE Internet Computing, 5(3):21-31, May/June 2001.

[BPS00] T. Bay, J. Paoli, and C. Sperberg-McQueen. *Extensible Markup Language (XML) 1.0*, W3C Recommendation, October 2000; <http://www.w3.org/TR/REC-xml>.

[DAML] *The DARPA Agent Markup Language Homepage*, <http://www.daml.org/>.

[DSC01] DAML Services Coalition (alphabetically A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, H. Zeng). *DAML-S: Semantic Markup for Web Services*. Proceedings of the International Semantic Web Working Symposium (SWWS). California, July 30-August 1, 2001.

[Fen01] Dieter Fensel. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer Verlag, ISBN: 3540416021, August 2001.

[HM01] J. Hendler and D. L. McGuinness. *DARPA Agent Markup Language*. IEEE Intelligent Systems, 15(6):72-73, 2001.

[HM02] P. Hallam-Baker and E. Maler. *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML)*. OASIS (Organization for the Advancement of Structured Information

Standards) Committee Working Draft, February 2002; <http://www.oasis-open.org/committees/security/docs/draft-sstc-core-27.pdf>.

[KPF01] Myong H. Kang, Joon S. Park, and Judith N. Froscher, *Access Control Mechanisms for Inter-Organizational Workflow*, 6th ACM Symposium on Access Control Model and Technologies (SACMAT), Chantilly, Virginia, May 3-4, 2001.

[LAS98] Ora Lassila. *Web Metadata: A Matter of Semantics*. IEEE Internet Computing, 2(4):30-47, July/August 1998.

[LHM84] Carl E. Landwehr, Constance L. Heitmeyer, and John McLean. *A Security Model for Military Message Systems*. ACM Transactions on Computer Systems, 2(3):198-222, August 1984.

[LS99] O. Lassila and R. Swick. *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation, February 1999; <http://www.w3.org/TR/REC-rdf-syntax/>.

[Notation3] *Notation 3*, <http://www.w3.org/DesignIssues/Notation3>.

[PICS] Platform for Internet Content Selection (PICS), <http://www.w3.org/PICS/>.

[PH01] Jeff Z. Pan and Ian Horrocks. *Metamodeling Architectures of Web Ontology Languages*. Proceedings of the International Semantic Web Working Symposium, California 2001.

[PH03a] Joon S. Park and Junseok Hwang. *Role-Based Access Control for Collaborative Enterprise in Peer-to-Peer Computing Environment*. 8th ACM Symposium on Access Control Models and Technologies, Como, Italy, June 2-3, 2003.

[PH03b] Joon S. Park and Junseok Hwang. *A Middleware Approach for SAINT (Secure, Automatic, Interoperable, and Transparent) Peer-to-Peer Content Services*. 8th IEEE Symposium on Computers and Communications, Antalya, Turkey, June 30-July 3, 2003.

[PKF01] Joon S. Park, Myong H. Kang, and Judith N. Froscher. *A Secure Workflow System for Dynamic Cooperation*. 16th International Conference on Information Security (IFIP/SEC 2001), Paris, France, June 11-13, 2001.

[PS00] Joon S. Park and Ravi Sandhu. *Binding Identities and Attributes Using Digitally Signed Certificates*. 16th Annual Computer Security Applications Conference (ACSAC), New Orleans, Louisiana, December 11-15, 2000.

[PSA01] Joon S. Park, Ravi Sandhu, and Gail-Joon Ahn. *Role-based Access Control on the Web*. ACM Transactions on Information and System Security (TISSEC), Volume 4, Number 1, February 2001.

[S2ML] *Security Services Markup Language (S2ML)*, <http://www.s2ml.org/>.

[SCFY96] R. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman. "Role Based Access Control Models," *IEEE Computer* 29 (2), February 1996.

[SWO] SemanticWeb.Org, <http://www.semanticweb.org/>.

[SWeb] *Semantic Web Activity*, <http://www.w3.org/2001/sw/>.

[WMA] W3C Metadata Activity, <http://www.w3.org/Metadata/Activity.html>.

[XKMS] *XML Key Management Specification*, <http://www.verisign.com/resources/gd/xml/xkms/xkmsv1-1.pdf>.

[XMLE] *XML Encryption Syntax and Processing*, <http://www.w3.org/TR/xmlenc-core/>.

[XMLS] *XML-Signature Syntax and Processing*, <http://www.w3.org/TR/xmldsig-core/>.

[XrML] Extensible Rights Markup Language (XrML), <http://www.xrml.org/>.