

Using Web Service Enhancements to Bridge Business Trust Relationships

Zhengping Wu and Alfred C. Weaver

*Department of Computer Science, University of Virginia, 151 Engineer's Way, P.O. Box 400740
Charlottesville, VA 22904
{zw4j,weaver}@cs.virginia.edu*

Abstract

With the development of web technology and distributed systems, online collaborations are becoming more common and more demanding. These collaborations require online business trust relationships among collaborating organizations. Online business trust relationships can protect the trust, integrity, and privacy of shared resources, which are the foundation for online business. Web services provide standard mechanisms to enable online interactions and further online collaborations. Yet security, privacy and trust-related protection mechanisms for web services need additional development. In an interconnected network environment, bridging extant business relationships to extend the business circle is a convenient and tempting way. Physical connections with proper privacy and security protections are required for bridging two autonomous organizations. Likewise, collaborating organizations need proper protection mechanisms for bridging extant business trust relationships among cooperating parties. These protection mechanisms must therefore ensure privacy and owner control in the entire process of bridging business trust relationships due to the subjectivity of the relationships. This paper describes an indirect trust establishment mechanism using web service enhancements to bridge and build new online business trust relationships from extant business trust relationships providing privacy protection and owner control simultaneously.

Keywords—*Web Service, Web Service Enhancements, Trust Establishment, Trust Management*

1. Introduction

Web services facilitate standards-based interactions between web servers, software agents, and end users, which enable inexpensive and dynamic online collaborations. But they do not address the business aspects of communication such as security, access control, business partner selection, service level agreement monitoring, and auditing – the activities that build business trust relationships between a consumer and a

provider of a business service and that will ultimately determine which services are used and which are not. The issues of trust and security are tightly bound in the minds of consumers. For example, a consumer would generally trust online services provided by Citibank to conduct online banking in a secure and responsible manner, because Citibank has a massive physical network of branches and has been in the financial market over one hundred years. On the other hand, a consumer probably wouldn't entrust his/her savings to a newly launched financial institution with no obvious connections to any legitimate business. The point here is that as web services begin to gain a foothold in electronic business, critical services will probably be limited to extensions of pre-existing business relationships with already trusted companies.

How can a business service provider engender new trust or transfer trust to a new consumer via an existing agent? And how can two companies establish a relationship of trust in order to provide and consume business services or share resources over web services? They have to negotiate an agreement in order to establish a trust relationship, and so they would almost certainly want to do the same for a trust relationship with web services. Although web service standards UDDI [1], WSDL [2], and SOAP [3] provide a scenario to publish, discover, and enable service consumption online, they say nothing about this trust relationship. Thus, although it would be possible to find a web service to use just by examining a UDDI registry, it is unlikely to be used prior to investigation of its reliability, reputation and trustworthy. To establish a trust relationship, the consumer and the service provider require a negotiation process. The negotiation process needs to exchange trust-related information between the two parties. The parties can exchange private attributes to build the trust relationship directly [4], or they can bridge pre-established relationships to build a new one via a common third party. Exchange of private attributes may put the privacy of the consumer and the service provider at risk. For example, a hacker may pretend to be a consumer to access useful information from the service provider. More seriously, a hacker can pretend to be a service provider to gather private attributes from consumers for malicious usage.

Bridging pre-established relationships is a more convenient and more secure alternative.

To provide additional functionalities for security, privacy and many other aspects of online interaction and collaboration, “web service enhancements” appear. Web service enhancements are a series of specifications describing security, privacy and other contexts applied to web services by several industrial practitioners. In this paper, we describe an indirect trust establishment mechanism using web service enhancements for bridging extant trust relationships (business trust relationships) to produce a new one. Using exchange of privileges obtained from a common third party (who has established trust relationships with both participants) avoids disclosure of any private attributes. Meanwhile this mechanism still allows free negotiation and trust agreement selection between the involved participants when subjective judgments have to be made.

2. Related work

Using indirect trust establishment mechanisms to bridge extant trust relationships is convenient and efficient to produce new trust relationships, but very few indirect trust establishment mechanisms have been designed for a web services environment. However, several types of indirect trust models and the corresponding trust establishment mechanisms have been proposed for service-oriented architectures [5], which are the foundations of web services. The trust relationships involved in the interactions between web services are enabled by separate authorities issuing security tokens [6], which certify the identities or other non-identity attributes for the consumers and/or providers of web services. There are two major types of trust models for these online trust relationships. One is the centralized model and the other is the distributed model.

In centralized trust models, a common trusted intermediary, called the “Trust Authority,” is used for establishing trust relationships between any two entities. However, it may be difficult to find an ideal central authority if the community of trustees is large and heterogeneous. If a trust authority is decided, token recipients are typically able to ascribe a sufficient level of trust to a security token because they can be confident of its origin. For example, they know and trust the authority that issued the token and can verify the token’s origin through cryptographic means. It is through the existing trust they have in the third party security token issuer that they are able to derive indirect trust for the holder of a security token created by the same issuer. The Privacy Enhanced Mail (PEM) certification [7] assumes that

everyone in the world trusts one ultimate authority to verify the identities of other certificate senders (an assumption we find unrealistic). The PEM model does not allow for multiple levels of trust within its certification hierarchy. Unlike PEM, the X.509 authentication framework and its variants for web services [8] follow a multiple trust authority structure. It postulates that everyone will obtain certificates from an official certifying authority (CA). The CAs are organized into a global hierarchy of certifying authorities. All users within a “community of interest” have keys that have been signed by CAs with a common ancestor in this global hierarchy, which forms a tree structure.

In distributed trust models, a static or dynamic web of trust is woven with less structured logical interactions between networks [9] compared with the centralized trust models. It is assumed that trust is transitive under certain contexts, because trust-related information can be propagated through one or more chains of trusted intermediaries in the networks. In the Pretty Good Privacy (PGP) system [10], an entity generates a public/private key pair. Each entity is responsible for acquiring the public key certificates needed and for assigning degrees of trust to their source. There is no common ancestor to act as the trust server for grouping the users within a community of interest. Instead, trust is propagated through chained structures formed by individual entities. When comparing X.509 with PGP, it has been pointed out that the most apparent difference is the architecture [11]. X.509 has hierarchic structures for professional or government organizations with liabilities, whereas PGP has anarchic structure based on informal relationships and undefined roles. Like the logic-based systems described in [12, 13], there is no partial trust (no degree of trust) in this kind of system; trust is either complete or nonexistent. In the solution proposed by Tarah et al. in [14], the degrees of trust from different entities could have conflicts, and the final degree of trust needs to be composed from different trust values.

Yet none of the above frameworks or systems can accommodate all trust models. However, all the trust models co-exist in the real world and all the trust models are used in daily life. So a flexible framework to accommodate all these trust models is desirable. Other associated questions remain open, e.g., the actual meaning of a trust value and how different trust values can be combined to yield a composite value. Privacy protection is another issue also. Users generally prefer to control their own private information. But none of the above mechanisms address privacy issues, which could lead to information leakage during the propagation of trust-related information in the trust establishment process. To solve these issues, we introduce an alternative method to

augment indirect trust establishment with negotiation to provide owner control in the process of bridging extant trust relationships. At the same time, to guarantee privacy protection, we use privileges granted by a common third party as a substitute for exchanging private attributes during the trust negotiation.

3. Indirect trust establishment

3.1. Web service enhancements

Web services use SOAP to exchange information over the network. SOAP is a lightweight method for exchanging structured information in a decentralized environment. XML (eXtensible Markup Language) is used in SOAP to define a flexible messaging framework that can exchange a message over a variety of underlying protocols. Although SOAP is the basic infrastructure for information exchange between web services, it does not provide any privacy and security mechanisms for the information exchanged. To provide privacy, security, trust, and other functionalities for web services, web service enhancements are proposed.

Among this set of enhancements, the web service security specification requires that an incoming access request message prove a set of claims such as name, public key, permission, capability, or an existing trust relationship to guarantee security. A web service indicates its requirements and other security related information in its policy document together with the privileges to be granted for the entities satisfying these requirements (e.g., a WS-Policy [15] file). If an access request arrives without having the required proof of claims, the service provider ignores or rejects the request. These claims are contained in security tokens. A security token is a representation of security-related information conveyed within the format of a SOAP message [15]. If an issuer cryptographically endorses a security token, the token is called a signed security token. A security token service (STS) is a web service that issues security tokens [6, 12]. That is, it makes assertions based on evidence that it trusts to whomever trusts it. To communicate trust, a security token service requires proof, such as a security token or a set of security tokens, and issues a new security token with its own trust statement (note that for some security token formats this can be just a re-issuance or co-signature). Another important related service is the attribute service. An attribute service is a web service that maintains attribute information about entities within a security domain.

With these services one entity can rely upon a second entity to execute a set of actions or to make a set of assertions about a set of subjects or scopes [16], which is called trust establishment. Trust relationships can be

established by exchanging private attributes or by bridging existing trust relationships; these techniques focus on owner control and utilizing extant trust relationships respectively. We propose a new indirect trust establishment mechanism to incorporate owner control into the process of bridging extant trust relationships.

3.2. Indirect trust establishment

The proposed indirect trust establishment mechanism uses a bridging protocol, which chooses a common third party as the anchor to bridge two extant trust relationships. An extant trust relationship is represented as a trust group element that includes a trust relationship name (T), a list of participants involved in this relationship, and a list of privileges (P) granted for that relationship. This trust relationship can be established via an online trust negotiation or a contract written on paper. In the bridging protocol, the common third party needs to discover any difference in privileges granted to the two participants in order to provide the two participants equal standing and the opportunity to make their own subjective decisions for the new trust relationship. Figure 1 shows the workflow of the bridging protocol to establish a new trust relationship between A and B using extant trust relationships between A and C, and B and C. The step numbers of the protocol correspond to the number labeled on the arrows in the figure.

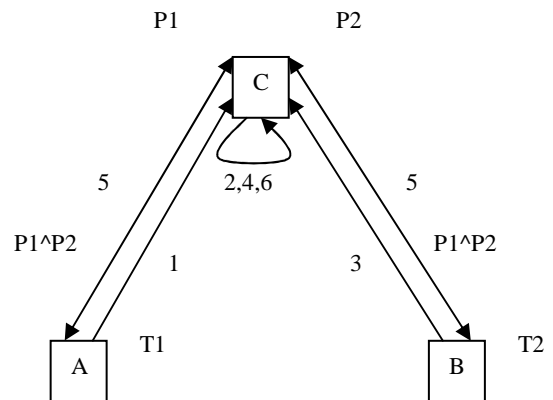


Figure 1. The bridging protocol

- (1) A sends to C a request to establish a trust relationship with B, using an extant trust relationship (T1) with C.
- (2) After C receives A's request, C waits for a similar request from B. If a time limit expires, C sends a fail message to A and exits its waiting state.
- (3) B sends to C a request to establish a trust relationship with A, utilizing its trust relationship (T2) with C.

(4) After receiving the requests from both A and B, C compares its granted privileges P1 and P2 for T1 and T2, and calculates the intersection of P1 and P2 ($P1 \wedge P2$).

(5) C sends $P1 \wedge P2$ to A and B, and asks whether they agree to build the new trust relationship based upon common privileges $P1 \wedge P2$.

(6) If both A and B respond with a positive answer, C sends confirmations to both A and B. Both A and B produce corresponding new policies for the newly established trust relationship, and create a new trust group element to represent this trust relationship. If either A or B rejects the privileges represented by $P1 \wedge P2$, C asks A and B to establish a trust relationship directly by mutual identity verification or private information exchange. The new trust group element contains a name for the new relationship, a list of participants within this trust relationship (here A and B), and a trust level decided by each participant. So the copies of the trust group element kept by A and B contain the same name and participant list, but have their own trust levels associated with this trust relationship.

This indirect trust establishment mechanism together with the bridging protocol resolves two problems described in section one. First, it introduces the negotiation process into the indirect trust establishment (bridging extant trust relationships), which assumes that every participant has the right to make its own decisions. Second, it prevents privacy leakage by exchanging privileges granted by the common third party instead of exchanging private attributes.

3.3. The common third party

How to find an appropriate common third party is also an issue. The participants A and B need to exchange partner information to find out who is appropriate for a common third party. As with most companies, A and B's partner lists should be expected to represent private information. Companies are generally unwilling to reveal their business partner lists to companies who have not already established a trust relationship. To solve this problem, we propose a complimentary protocol below to find all common third parties.

(1) A and B send requests to all their trusted partners respectively, to ask if they are willing to act as the common third party.

(2) Only participants who are both A and B's trusted partner receive requests from both A and B. If it agrees to act as the common third party, it sends a message indicating its willingness to A and B.

(3) From all the candidates who send back their willingness, A and B choose one as the common third party.

Either A or B can express its willingness to use one of the commonly trusted parties. The potential partner either agrees or starts a brief negotiation to establish the common third party before the indirect trust establishment process.

4. System architecture

Figure 2 illustrates the system architecture. With the help of the proposed bridging protocol, trust domains can establish new trust relationships by extending current trust boundaries more smoothly. In this system architecture, the STSs are the main portals for interactions across trust domains. The STSs are in charge of issuing and exchanging security tokens, which contain critical information such as identities, privileges and trust-related information. A policy repository is used to store and retrieve policy requirements and the corresponding privilege information used by the bridging protocol. The negotiation engine controls the whole process of information exchange and negotiation step by step. Each trust domain has an entire deployment of this system. But according to the bridging protocol and indirect trust establishment process, the functionalities used by participants A, B and the common third party C are different.

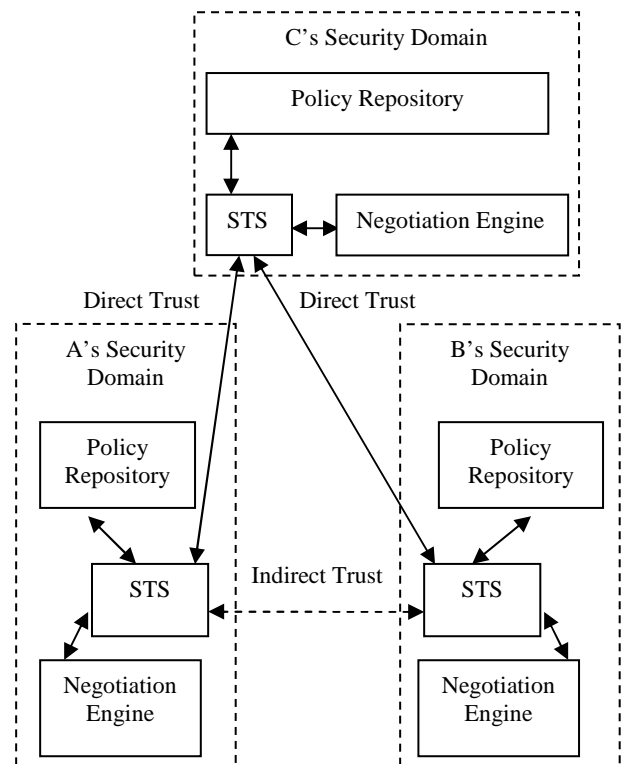


Figure 2. The Architecture of the Indirect Trust Establishment Mechanism

5. Implementation

Our implementation of this system architecture utilizes the Microsoft .NET framework. We also use a set of functionalities provided by the Web Services Enhancements (WSE) [18] toolkit to handle token issuance, token verification, policy description, and policy management.

We introduce a trust group element to represent an established trust relationship. But a trust group element is represented in different formats in security tokens and the policy repository. In a security token, the trust group element contains a name for the relationship together with information representing the two domains between which the trust relationship is established. In the policy repository, every trust group element not only contains the information expressed in security tokens but also is associated with a set of privileges granted by another party. Figure 3 illustrates these two different formats of a trust group element in a security token and in the policy repository.

Two types of security tokens are used with different purposes to convey identities, credentials, trust group elements and other trust related information between service modules. We use username tokens to facilitate the interactions between service modules within the same security domain since it has little overhead and is easy to extend; we use SAML [17] tokens to enable interactions between security domains since SAML is a recognized standard for interoperability among different platforms. Figure 4 compares these two token types. We also follow the WS-Policy [15] specification to describe the requirements and privilege information for our bridging protocol and indirect trust establishment mechanism.

We provide an interactive trust establishment framework with a special user interface to control the progress of indirect trust establishment process. Meanwhile, if a domain administrator wants to automate the entire indirect establishment process, we allow him/her writing a script in advance to describe an appropriate policy for every step in the bridging protocol. Then the trust establishment framework can consult the script automatically to determine what to do next in the indirect trust establishment process.

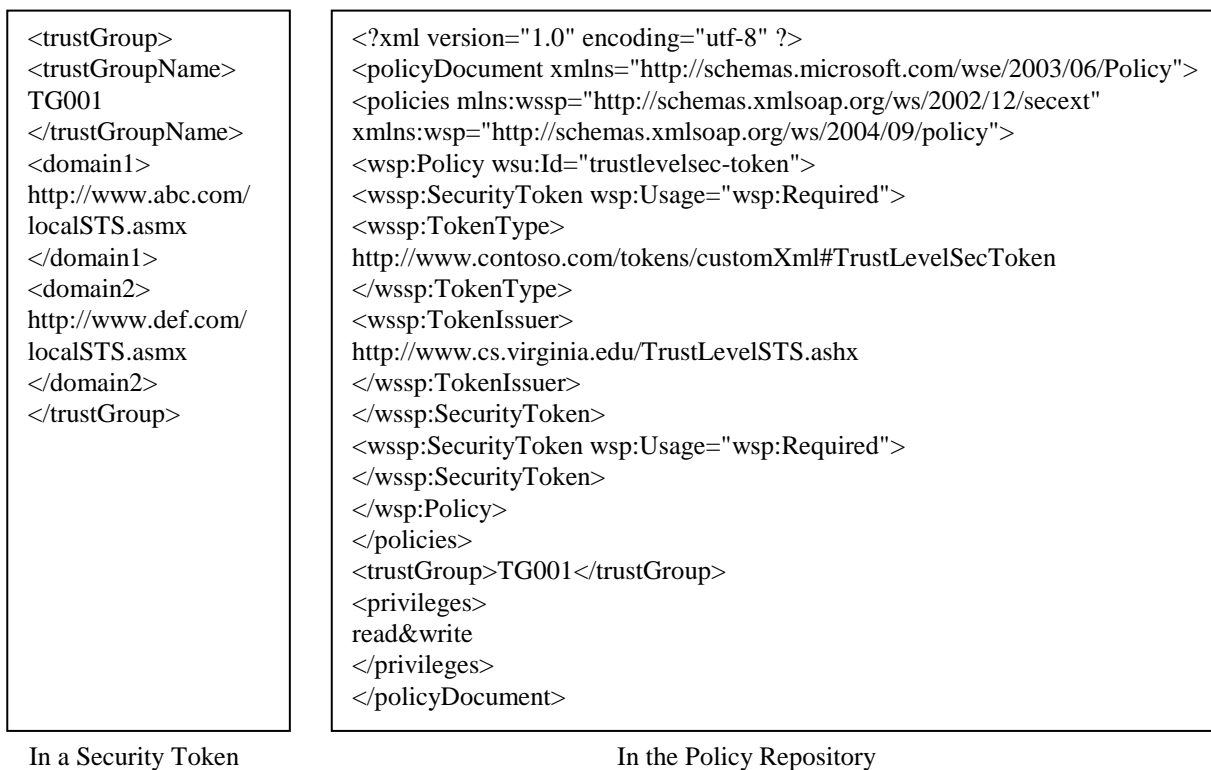


Figure 3. A Trust Group Element in a Security Token and in the Policy Repository

<pre> <UsernameToken> <CreateAt>01/20/2006 08:00:00 – 05:00 </CreateAt> <ExpireAt>01/20/2006 17:00:00 – 05:00 </ExpireAt> <UserID>123</UserID> <TokenIssuer> http://abc.com/TrustSTS.asmx </TokenIssuer> <TrustGroup>Name="TG001" Domain1="http://abc.com" Domain2="http://def.com" </TrustGroup> </UsernameToken> </pre>	<pre> <saml:Assertion Version="2.0" ID="ABC" IssueInstant="timestamp"> <saml:Issuer>http://abc.com/TrustSTS.asmx </saml:Issuer> <saml:Conditions NotBefore="01/20/2006 08:00:00 - 05:00:00" NotOnOrAfter="01/20/2006 17:00:00 - 05:00:00"> <saml:Subject> <saml:NameID>123</saml:NameID> </saml:Subject> <saml:Attribute> Name="TrustGroup" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname- format:basic" <samlAttributeValue>TG001</samlAttributeValue> Domain1="http://abc.com" Domain2="http://def.com" </saml:Attribute> </saml:Assertion> </pre>
Username Token	SAML Token

Figure 4. Comparison of Username Token and SAML Token

6. Discussion

There are several advantages of the proposed bridging protocol and indirect trust establishment mechanism. The most obvious advantage of indirect trust establishment mechanisms is that each entity (an individual user or a trust domain) can use extant trust relationships to build new ones. This process can be totally automatic based on the degrees of trust assigned to extant trust relationships, which relieves the burden on domain administrators to build new trust relationships manually; alternatively, the process can be augmented by a lightweight negotiation (described in this paper) to provide owner control to protect private information. If the policies and strategies required for the negotiation protocol and the corresponding supplementary information are set in advance (e.g., written in a script file), the lightweight negotiation process can also be performed automatically. So this indirect trust establishment mechanism provides owner control without increasing users' or administrators' burdens.

Compared to other indirect trust establishment solutions, this mechanism does not limit a trust relationship to binary trust (complete trust or no trust) as described in [12, 13]; it assigns trust levels to the newly built trust relationship to guarantee the flexibility to distinguish different levels of trust granted by different

participants. This mechanism also does not use a pre-set formula to calculate the ultimate degree of trust as in [14], when different extant relationships imply different or conflicting degrees of trustworthiness for the new trust relationship; instead, it provides an alternative method to allow the participating entities to do a lightweight negotiation to permit a subjective decision on the ultimate level of trust for the new trust relationship.

Because no private attributes are used in the implementation, user's privacy is protected; because the policies that describe users' preferences and decision-making criteria can be set (described in a script) before the negotiation, this bridging protocol and indirect trust establishment mechanism can be totally automated, thus achieving effectiveness and efficiency simultaneously. The only limitation lies in the process to find an appropriate common third party. Because both entity A and B in the bridging protocol need to query all potential common third parties, scalability will be an issue if both of them have large numbers of trusted partners. In our implementation, we do not encounter this difficulty.

7. Conclusion

In this paper we described an indirect trust establishment mechanism using a bridging protocol

augmented with lightweight negotiation to achieve owner control and privacy protection simultaneously. Our research motivation comes from the business relationship expansion model in the real world and the risk of privacy leakage when exchanging private attributes to negotiate a new trust relationship. Our new indirect trust establishment mechanism is an alternative method with these advantages:

- It introduces the negotiation process into the indirect trust establishment (bridging extant trust relationships), which assures that every participant has owner control over the decision-making process for new trust relationships.

- It prevents privacy leakage by exchanging privileges granted by the common third party instead of exchanging private attributes.

Our future research will focus on the topological impact and scalability of this bridging protocol and indirect trust establishment mechanism and its application to privilege delegation.

8. References

[1] Bob Atkinson, et al., “UDDI Spec Technical Committee Specification,” October 2003.
<http://uddi.org/pubs/uddi-v3.0.1-20031014.htm>

[2] Erik Christensen, et al., “Web Services Description Language (WSDL) 1.1,” March 2001.
<http://www.w3.org/TR/wsd1>

[3] Don Box et al., “Simple Object Access Protocol (SOAP) 1.1,” May 2000.
<http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

[4] Z. Wu, A. C. Weaver, “Dynamic Trust Establishment with Privacy Protection for Web Services,” *Proc. 2005 IEEE International Conference on Web Services*, pp. 811 – 812.

[5] M. P. Papazoglou and D. Georgakopoulos, “Service-oriented Computing,” *Communications of the ACM*, Vol. 46, No. 10, October 2003, pp. 25 – 28.

[6] A Joint White Paper from IBM Corporation and Microsoft Corporation, “Security in a Web Services World: A Proposed Architecture and Roadmap,” <http://msdn.microsoft.com/library/en-us/dnwssecur/html/securitywhitepaper.asp>, April 2002.

[7] Kent, S. “Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management,” Internet Report, RFC 1422, February 1993.

[8] Phillip Hallam-Baker, et al., “Web Services Security X.509 Certificate Token Profile,” March 2004.
<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf>

[9] Nathan Dimmock, “Using Trust and Risk in Role-Based Access Control Policies,” *Proc. 2004 Symposium on Access Control Models and Technologies*, June 2004.

[10] P.R. Zimmermann, “The Official PGP User’s Guide,” MIT Press, 1995.

[11] A. Josang, “The right type of trust for distributed systems,” *Proc. 1996 New Security Paradigms Workshops*.

[12] P. V. Rangan, “An Axiomatic Theory of Trust in Secure Communication Protocols,” *Computers and Security*, Vol. 11, 1992, pp. 163 – 172.

[13] M. Abadi, “Logic in Access Control,” *Proc. of the 18th Annual IEEE Symposium on Logic in Computer Science*, June 2003, pp. 228 – 233.

[14] A. Tarah, Ch. Huitema, “Associating Metrics to Certification Paths,” *Proc. 2nd European Symposium on Research in Computer Security*, 1992, pp. 175 – 189.

[15] Siddharth Bajaj, et al, “Web Services Policy Framework (WS-Policy),” September 2004.
<http://specs.xmlsoap.org/ws/2004/09/policy/ws-policy.pdf>

[16] Steve Anderson, et al., “Web Services Trust Language (WS-Trust),” May 2004.
<http://msdn.microsoft.com/ws/2004/04/ws-trust/>

[17] Scott Cantor, et al., “Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0,”
<http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>, March 2005.

[18] Web Services Enhancements, “Major Features of the Web Services Enhancements,”
<http://msdn.microsoft.com/library/en-us/wse/html/3c55ed70-6b66-4620-bba7-59c5b2f2191d.asp>