Australian Journal of Basic and Applied Sciences, 7(6): 27-33, 2013
ISSN 1991-8178

Formal Reference Ontology for Securing Data Objects to Support Semantic Web Services (SWS)

[1]Saravanan Muthaiyah, [2]Seethaletchumy Thambiah

[1]Faculty of Management, Multimedia University,Cyberjaya,Malaysia
[2]Faculty of Management, Multimedia University,Cyberjaya,Malaysia

Abstract: Semantic Web Services (SWS) utilize powerful semantic technologies to search, acquire and package services that are machine understandable. However, the true potential of SWS can only be realized if transacted data objects are secure,confidential and reliable. Thus, a meta model for security policies and frameworks for SLAs (Service Level Agreements) has to be in place to offer reliable SWS to Semantic Web user community. Security policies can be interpreted in different ways as it is implemented in different ways by VO today. Multiple security policies of VO would require a reference lookup i.e. a RO which is truly the most efficient method of achieving interoperability of respective VO security policies. The aim of this paper is to introduce a RO for security policy specifications i.e. SPRO that can be shared across all VO participating in the Semantic Web. In this paper we also introduce an approach to secure dynamic packaging and data hiding to improve the confidentiality, authentication and integrity of transacted data objects.

Key words: Semantic Web Service, Data hiding, Semantic Web, Ontology, Semantic Information Integration, Dynamic Packaging, Steganography.

INTRODUCTION

The tourism industry today, serves a huge market that includes thousands of e-tourism and e-travel sites on the web. Given such ubiquity, key players in this industry such as airlines, hoteliers, car rental companies, leisure suppliers and travel agencies must take preventive measures to secure confidential data of their customers. Dynamic packaging is the technique of combining different travel components, bundling and it's pricing in real time and on-the-fly in response to the request of a potential traveller or a booking agent. It's a web service that consolidates travel dates, budget, payment, discount vouchers, reservation and method of payment data objects to provide the best travel alternatives for a traveller. RDF (Resource Description Framework) provides machine understandable documents and helps in searching and cataloguing for such data.

The RDF structure has three parts i.e. resources, properties and statements. Statements include subjects, objects and predicates. Since, security cannot be viewed in isolation, an end-to-end security mechanism is needed for the Semantic Web technology stack which includes a protocol stack (the lowest layer), XML (Bishop, M., 2002), RDF (Fowler, J., *et al*., 1999) and OWL (Fowler, J., *et al*., 1999).

As such RDF has to be secured in order to secure SWS. WS-Security specification makes recommendations on XML Signature and XML Encryption, to ensure the integrity and confidentiality of SOAP messages. Digital credentials and their relationship to Trust Semantics with SOAP messages are also being specified. The term "security policy" can be interpreted in many different ways. A security policy is a statement of what is allowed, and not (Giunchiglia, F.a.I.Z., 2002). To effectively manage security policies we should produce compatible policy representations. Multiple interacting security policies require semantics to be manipulated and SPRO is an efficient means for achieving this. There are many aspects of security policy that need to be reconciled between the global and local environments. A complete taxonomy is needed so that the reconciliation can be performed quickly. Literature indicates that a comprehensive ontology for security policy doesn't exist. Therefore one had to be created for the VO and the RO (Muthaiyah, 2006b).Security policies can be divided into authentication, authorization, integrity, access control, non-repudiation and confidentiality policies individually or a combined version of all. This paper focuses on the *authorization security policy* for SPRO.

1.1 Motivation And Scope:

With the term "Semantic" we mean the formal (and thus unambiguous) description of some particular object (more in section 2), which is subject to automated ontology-based reasoning. Within the context of the Reference Ontology, these objects are mainly the data handled by the services and the services themselves. Semantic descriptions within SOAs allow reasoning tools to automate tasks. More specifically, semantics help in the following ways:

- Formally and unambiguously define the data models and processes underlying the system;
- Allow automated discovery and composition of services;

Corresponding Author: Saravanan Muthaiyah, Faculty of Management, Multimedia University,Cyberjaya,Malaysia
E-mail: seethaletchumy@mmu.edu.my

- Automatically resolve data and process mismatches, easing integration and improving interoperability;
- Ease the process of service ranking, negotiation and contracting.

The scope of this document is therefore to provide an ontology that formally describes the different elements comprising a SSOA in order to achieve the above objectives.

1.2 Audience:

The target audience for this document extends that of the SOA RM; however, we provide an exhaustive list in order to keep the document self-contained as illustrated in Figure 1 below:

- Architects and developers designing, identifying or developing a system based on the Service Oriented Architectures;
- Standards architects and analysts developing specifications that rely on Service Oriented Architecture concepts;
- Decision makers seeking a "consistent and common" understanding of Service Oriented Architectures;
- Users who need a better understanding of the concepts and benefits of Service Oriented Architectures;
- Academics and researchers that are researching within the Semantic Web and Semantic Web Service communities;
- I.T. consultants that provide businesses with support on Semantic technologies and SOAs in general.

Business Domain	Business Domain Specific Extensions	Various
Management	Distributed Management	WSDM, WS-Manageability
	Provisioning	WS-Provisioning
Security	Security	WS-Security
	Security Policy	WS-Security Policy
	Secure Conversation	WS-SecureConversaion
	Trusted Message	WS-Trust
	Federated Identity	WS-Federation
Portal & Presentation	Portal and presentation	WSRP
	Asynchronous Services	ASAP
Transaction and Business Process	Transaction	WS-Transactions,WS-Coordination,WS-CAF
	Orchestration	BPEL4WS, WS-CDL
Messaging	Events and Notification	WS-Eventing,WS-Notification
	Multiple message sessions	WS-Enumeration,WS-Transfer
	Routing addressing	WS-Addressing, WS-MessageDelivery
	Reliable messaging	WS-Reliable Messaging,WS-Reliability
	Message packaging	SOAP,MTOM
Metadata	Publication and discovery	UDDI,WSDL
	Policy	WS-Policy,WS-Policy Assertions
	Base service and message description	WSDL
	Metadata retrieval	WS-Metadata Exchange

Fig. 1: Web Service Protocol Stack (Source: W3C)

2. Securing Data Objects:

Dynamic packaging technology is capable to allow travellers to choose customized trips that combine customer preferences with flights, car rentals, hotel, and leisure activities all within one quotation. Dynamic packaging enables consumers or the booking agents to build customized itinerary by assembling multiple components of their choices and complete the transaction in real-time. It is based on an individual consumer's request, including the ability to combine multiple travel components like flights, hotels, car rentals that provide a single fully priced package, requiring only one payment from the consumer and hiding the pricing of individual components within a few seconds. The products available to customer can then be stored in local inventories or external sources. The architecture of this system is composed of three layers i.e. the integration layer, reasoning layer, and the packaging layer. The integration layer includes all data sources e.g. structured, semi-structured and unstructured objects. The information is then wrapped and transferred into ontological sources where the instance generator allows inferences of OWL instances by querying the knowledge-base. The reasoning layer uses a rule engine to fire rules which is supported by an external reasoned i.e. Racer Pro (Halevy, A. Y., *et al*. 2004) using SWRL (Hovy, E., 1998). The dynamic packaging layer is responsible for reading the packaging rules specifications and generating valid packages, i.e. travel packages that comply with the packaging rules. The RDF data is then labelled using security authentication via steganography methods described in detail in the following section.

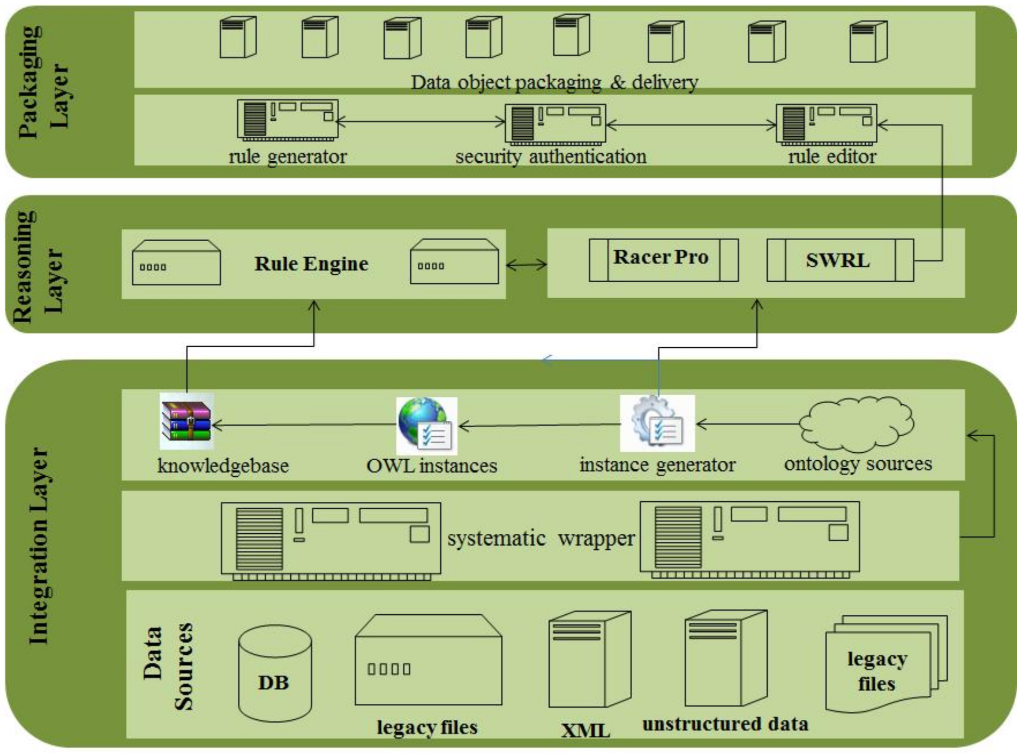

Fig. 2: Dynamic Packaging Architecture

Data object packages are automatically reasoned by this layer. The architecture is supported by a dynamic rule and security authentication engine in the packaging layer. The configuration of the reasoning layer involves the following activities: rule development (where the rule designer defines packaging rules using the rule editor), a reasoning component that accesses the rule repository and the creation of packaging rules. Packaging rules are codified and stored in an integrated repository, providing a central point for definition and change, which can later drive dynamic package construction. The construction of packages may also involve querying the knowledge base. This is especially important when a dynamic package has already been put together according to the packaging rules and it is necessary to add information describing each product. This information can easily be obtained from the knowledge base. The security engine in the packaging layer secures and labels the classification of RDF documents where the steganography technique is applied. This would mean applying access controls at different levels including resources, properties and statements. The end result is to declare which documents are confidential and which are not.

3. Reference Ontology To Securing Data Objects:

This Reference Ontology for Semantic Service Oriented Architectures is an abstract framework for understanding significant entities and relationships between them within a Semantically-enabled Service-Oriented environment. It may be leveraged for the development of related standards or specifications supporting that environment, as well as guiding efforts to realize concrete solutions. This Reference Ontology builds on the OASIS Reference Model for Service Oriented Architecture (SOA-RM) and combines it with the key concepts of semantics that are relevant for Semantically-enabling Service Oriented Architectures. A reference model is not directly tied to any standards, technologies or other concrete implementation details. It does seek to provide a common understanding that can be used unambiguously across and between different implementations. The relationship between this Reference Ontology, the SOA Reference Model, and particular architectures, technologies and other aspects of SOA is illustrated in Figure 1.Just as the SOA-RM, this reference ontology focuses on the field of software architecture. The concepts and relationships described may apply to other "service" environments; however, this specification makes no attempt to completely account for use outside of the software domain.

Although Service Oriented Architectures (SOAs) have gathered a lot of attention within business organizations, for a long time there was no clear understanding of what an SOA precisely is. As a result reference models have been published to define SOA; we note particularly the OASIS SOA Reference Model (Bishop, M., 2002). However, with the emergence of **Semantic Web** technologies, in particular **Semantic Web Services (SWSs)**, new breeds of SOAs are being developed, namely **Semantic Service Oriented Architectures**

(SSOAs). SSOAs use semantic technologies to advance solutions to problems by which SOAs are limited. They provide a means for further automation for service consumers' tasks, particularly service discovery, selection, composition and execution, as well as easing general interoperability issues between services. In order to use the semantic descriptions present in a SSOA to automate such SOA features, a set of platform services that provide this automation functionality are required within the SSOA. These services are collectively termed a **Semantic Execution Environment (SEE)** for Semantic Web Services, with a SEE being at the core of a SSOA. There are a number of different implementations of SEEs currently under development in the research community, which have some common features. Thus the purpose of this document is to define an extended reference model for SSOAs, as supported by SEEs. This model will be defined formally using an ontology. The aim of this ontology is to provide a point of reference formally specified so that it can support the definition and development of SSOAs.

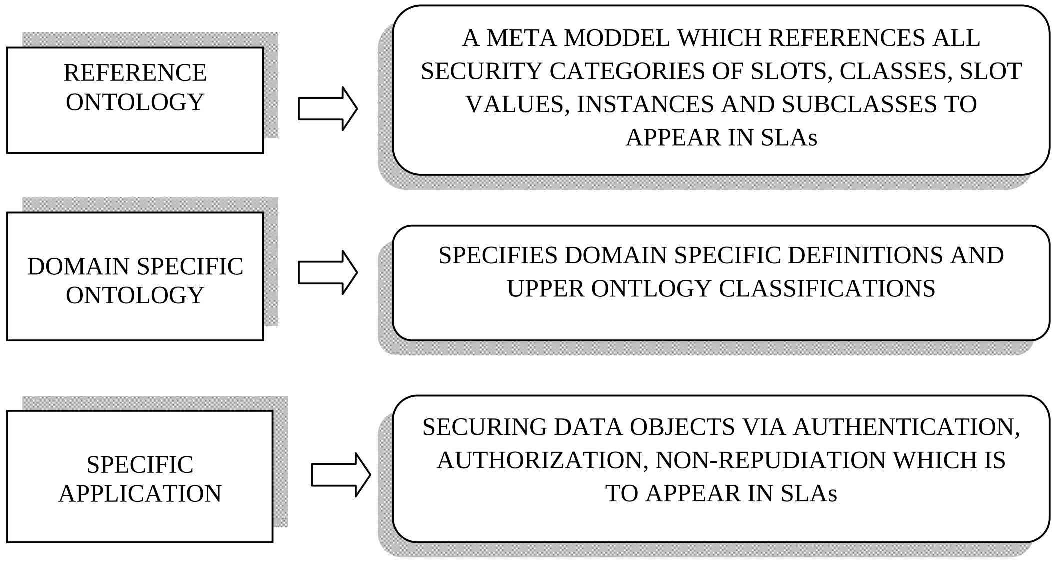
REFERENCE ONTOLOGY
A META MODDEL WHICH REFERENCES ALL SECURITY CATEGORIES OF SLOTS, CLASSES, SLOT VALUES, INSTANCES AND SUBCLASSES TO APPEAR IN SLAs
DOMAIN SPECIFIC ONTOLOGY
SPECIFIES DOMAIN SPECIFIC DEFINITIONS AND UPPER ONTLOGY CLASSIFICATIONS
SPECIFIC APPLICATION
SECURING DATA OBJECTS VIA AUTHENTICATION, AUTHORIZATION, NON-REPUDIATION WHICH IS TO APPEAR IN SLAs

Fig. 2.1: Relationship of the Reference Ontology to Other SOA Specifications and Standards

Figure 1-1 depicts how the Reference Ontology relates to other pieces of work within the SOA community. The figure is derived from Figure 1 in the SOA Reference Model document (Bishop, M., 2002) and introduces the Reference Ontology alongside the Reference Model element. The Reference Ontology presented in this document is a further step towards formalization of the Reference Model but also accommodates the extensions associated with Semantic Web Services resulting in Semantic SOAs. Since the start of this work, the SOA-RM committee have also started work on a Reference Architecture, which also aims at further formalisation of the reference model, but we consider ontologisation central to the semantics-based approach and diverge. Indeed when we say Reference Architecture we shall refer to a reference architecture for SEEs, not to the SOA Reference Architecture. Furthermore when we say Concrete Architectures we refer to implementations of semantics-enabled SOAs such as WSMX (Fowler, J., *et al.*, 1999), IRS III (Fowler, J., *et al.*, 1999) and METEOR-S (Giunchiglia, F.a.I.Z., 2002). The Related Models in Figure 1 include, for us, the Web Service Modeling Ontology (WSMO) (Gruber, T.R., 1993), Semantic Annotations for WSDL and XML Schema (SAWSDL) (Kalfoglou, Y.a.W.M.S., 2003) the Web Ontology Language for Services (OWL-S)[1] (Kashyap, V.a.A.P.S., 1996) and the Semantic Web Services Ontology (SWSO) (Li, L., *et al.*, 2005). Patterns fulfill the same role in Semantic- as in pre-Semantic- SOA, which is to say that they define more specific categories of service-oriented designs. The Protocols and Profiles (those considered as part of the related work) are the same as for classical SOAs. However, with respect to Specifications and Standards, we further take into account emerging Semantic Web Languages such as the OWL, RDF and RIF standards from W3C, and the WSML and SWSL de facto standards. These "standards" play a very important role since they are the pillars of Semantic Technologies. The Input features (Requirements, Motivation and Goals) are the same as for SOAs, with the addition that we have more emphasis on automation, as stated earlier.

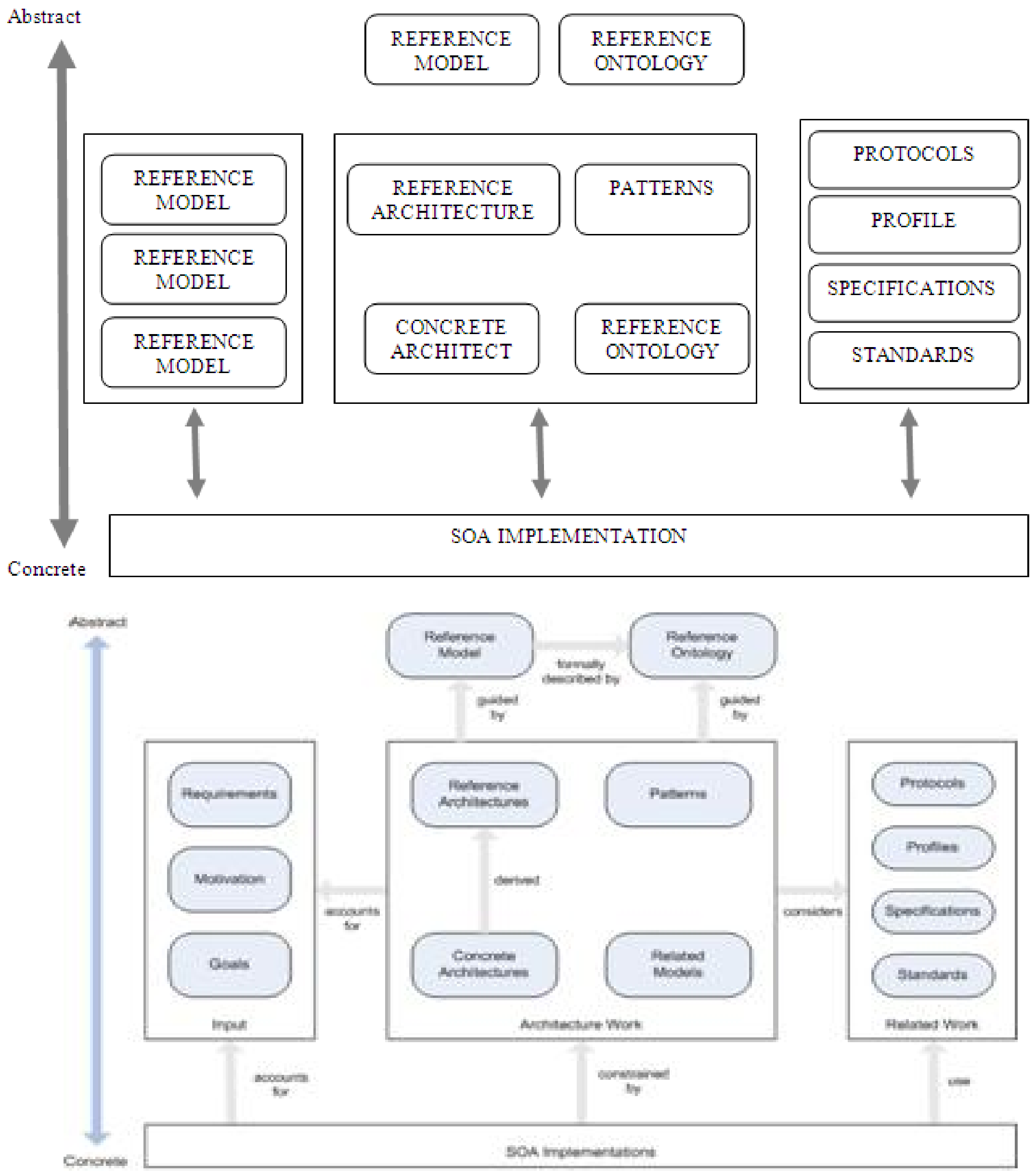

Fig. 2.2: Relationship of the Reference Ontology to Other SOA Specifications and Standards

4. Methodology:

The term "security policy" has many different meanings and can be interpreted in many different ways. A security policy is a statement of what is allowed, and what is not allowed (Bishop, 2002).To effectively manage security policies we must be able to produce compatible policy representations. The existence of a large number of representation methods leads to the conclusion that security policies, even if semantically compliant, can be represented in ways that differ in terms of formalism, structure, and hierarchy, thus raising obstacles to their reconciliation. Multiple interacting security policies require semantics to be managed and manipulated. The SPRO is an efficient means for achieving this. There are many aspects of security policy that need to be reconciled between the global and local environments. A complete taxonomy is needed so that the reconciliation can be performed quickly. Literature indicates that a comprehensive ontology for security policy doesn't exist. Therefore one had to be created for the VO and the RO (Muthaiyah, 2006b).

Example 1 – Authorization

Pseudo code	Actual Code
<!-- AUTHORIZATION This section sets the authorization policies of the application. You can allow or deny access to application resources by user or role. Wildcards: "*" mean everyone, "?" means anonymous (unauthenticated) users. -->	<authorization> <deny users="?" /> <!-- Allow all users --> <!-- <allow users="[comma separated list of users]" roles="[comma separated list of roles]"/> <deny users="[comma separated list of users]" roles="[comma separated list of roles]"/> --></authorization>

Example 2 – Authentication

Pseudo code	Actual Code
<!-- AUTHENTICATION The authentication policy of the application uses "Forms" mode "Windows" IIS performs authentication (Basic, Digest, or Integrated Windows) Authentication is performed via a centralized authentication service provided a single logon --> <authentication mode="Forms">	<forms loginUrl="Login.aspx"> </forms>

Table 2: Merging two SPRO policy data elements

	Security Policy – VO (Authentication) *Before Mapping*	Security Policy – RO (Authorization) *Before Mapping*	Compatible Security Policy (SPRO) (Authorization) *After Mapping*
SCENARIO A	A=Identifier +Password <? xml version="1.0" ?> <Final_Policy > <Entity> <Type/> <Identifier>UserId</Identifier> <Password> UserPassword </Password>	A=Identification with token ID <?xml version="1.0" ?> <Final_Policy > <Entity> <Type/> <Identifier>UserId</Identifier> <TokenId>Token</TokenId>	A=Password ticket <? xml version="1.0" ?> <Final_Policy > <Entity> <Type/> <Identifier>UserId</Identifier> <Password Ticket> UserPassword </Password Ticket>
SCENARIO B	B=X.509 Certificate <?xml version="1.0" ?> <Final_Policy > <Entity> <Type/> <Identifier>UserId</Identifier> <Certificate>X.509</Certificate >	B=X.509 token <?xml version="1.0" ?> <Final_Policy > <Entity> <Type/> <Identifier>UserId</Identifier> <Token>X.509</Token>	B=X.509 Certificate & token <?xml version="1.0" ?> <Final_Policy > <Entity> <Type/> <Identifier>UserId</Identifier> <CertificateToken>X.509 </CertificateToken>

Security policies can be divided into authentication, authorization, integrity, access control, non-repudiation and confidentiality policies individually or a combined version of all. This paper focuses on the *authorization security policy* for SPRO. Authorization verifies **what you are authorized to do**. For example, you are allowed to login into your Unix server via ssh client, but you are not authorized to browser /data2 or any other file system. Authorization occurs after successful authentication. Authorization can be controlled at file system level or using various application level configuration options such as chroot(2). Usually, the connection attempt must be both authenticated and authorized by the system. You can easily find out why connection attempts are either accepted or denied with the help of these two factors. Authentication verifies **who you are**. For example, you can login into your Unix server using the ssh client, or access your email server using the POP3 and SMTP client. Usually, PAM (Pluggable Authentication Modules) are used as low-level authentication schemes into a high-level application programming interface (API), which allows programs that rely on authentication to be written independently of the underlying authentication scheme.

Table 2 above shows two scenarios A and B where the hybrid model is applied to bridge the security definitions. "Final policy" data labels for authentication in scenario A are "UserId" and "UserPassword" and authorization data labels are "UserId" and "Token". SPRO shows the mapped data labels, which are "UserId" and "UserPassword". The identifier remains as "UserId" but Password and TokenId becomes "Password Ticket". In scenario B, "final policy" data labels for authentication are "UserId" and "X.509" (Certificate) before mapping. After mapping SPRO shows the new data labels as "UserId" and "X.509" (Certificate Token).

Merge is completed without any discrepancies and SPRO establishes a common global policy for authorization, which is agreed by VO and RO. Heterogeneity in the data labels are reconciled to arrive at the common policy.

Conclusion:

This paper provides an end-to-end framework for ontology mediation with emphasis on the hybrid model. Empirical test validate the hypothesis that the SRS scores of this model does produce more reliable, precise and relevant results compared to pure syntactic matching systems. It also highlights why syntactic and semantic measures are good for ontology mediation. Empirical tests also show that SRS scores and HCR scores have a strong positive correlation. A detailed process methodology is also introduced to show how the model is implemented. Our model does not incorporate context changes in measuring similarity and we intend to research how context changes can be incorporated into our hybrid model. This would be our future research direction.

REFERENCES

Bishop, M., 2002. *Computer security : art and science*.

Fowler, J., P. Brad, N. Marian, & B. Bruce, 1999. Agent-Based Semantic Interoperability in InfoSleuth. SIGMOD Record, 28(1): 60-67.

Fowler, J., *et al*., 1999. Agent-Based Semantic Interoperability. *InfoSleuth*.

Giunchiglia, F.a.I.Z., 2002. *Making Peer Databases Interact - A Vision for an Architecture Supporting Data Coordination.* Paper presented at the Conference on Information Agents, Madrid.

Gruber, T.R., 1993. Toward principles for the design of ontologies used for knowledge sharing.

Halevy, A. Y., *et al*. 2004. The Piazza Peer Data Management System. IEEE Transactions on Knowledge and Data Engineering, 16(7): 787-798.

Hovy, E., 1998. *Combining and standardizing large scale, practical ontologies for machine translation and other uses.* Paper presented at the 1st International Conference on Language Resources and Evaluation (LREC), Granada, Spain

Kalfoglou, Y.a.W.M.S., 2003. Ontology Mapping:state of the art. Knowledge Engineering Review, 18(1): 1-31.

Kashyap, V.a.A.P.S., 1996. Semantic and Schematic Similarities Between Database Objects: A Context-based Approach. VDLB Journal, 5(4): 276-304.

Li, L., Y. Yang, and B. Wu, 2005. *Agent-based ontology mapping towards ontology interoperability.* Paper presented at the Australian Joint Conference on Artificial Intelligence (AI'05), Sydney, Australia.

Madhavan, J., *et al*., 2002. *Representing and Reasoning about Mappings between Domain Models.* Paper presented at the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence, Edmonton,Alberta,Canada.

Maedche, A., *et al*., 2002. *MAFRA-A MApping FRAmework for Distributed Ontologies.* Paper presented at the 13th International Conference, Knowledge Engineering and Knowledge Management, Siguenza, Spain.

Missikoff, M., F. Schiappelli, and F. Taglino, 2003. *A Controlled Language for Semantic Annotation and Interoperability in e-Business Applications.* Paper presented at the ISWC 03, Sanibel Island,Florida.

Muthaiyah, S.a.L.K., 2006a. *Dynamic Integration and Semantic Security Policy Ontology Mapping for Semantic Web Services (SWS).* Paper presented at the First IEEE International Conference on Digital Information Management (ICDIM), Bangalore, India.

Muthaiyah, S.a.L.K., 2006b. *Virtual Organization Security Policies: An Ontology-based Mapping and Integration Approach.* Paper presented at the The Second Secure Knowledge Management Workshop (SKM) 2006, Brooklyn, New York.

Nilsson, M.R.G.a.N.J., 1987. *Logical Foundations of Artificial Intelligence.* San Francisco, CA: Morgan Kaufmann Publishers.

Noy, N.F.a.M.A.M., 2000. *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment.* Paper presented at the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, Austin,Texas.

Park, J.a.S.R., 2004. Information Systems Interoperability: What Lies Beneath? , 22(4): 595-632.

Software, M.G.a.M.P., Levenshtein Distance, in Three Flavors. from http://www.merriampark.com/ld.htm, http://www.merriampark.com/ld.htm#DEMO