Web Service Security Policy Matching Based on Semantics

Amira Abdelatey, Mohamed Elkawkagy, Ashraf Elsisi, Arabi Keshk
Faculty of Computers and Information/Computer Science, Menofia University, Egypt

Abstract: *An improved matching algorithm of Web service security concerns is proposed in this paper. It performs matching requirement and capability security concerns of both consumer and provider. WS-SP specifies the security requirements or capabilities of different web service participants. Security requirement/capability of a participant is one of the most important non-functional properties of a web service. The security concerns addressed within this paper emphasis on the security of a message (integrity and confidentiality) transmitted between the two participants. The proposed matching algorithm states matching of a simple policy and complex policy of WS-SP. In addition, a generalized matching algorithm is introduced to get the best-matched provider from a list of available providers for consumer requirements.*

Keywords: *Web Service Security concerns, Web Service Non-Functional Properties, Web Service matching, ontology matching, Semantics, SOA Message Security.*

Received June 16, 2016; Accepted August 9, 2017

1. Introduction

Nowadays, SOA has considered as a Middleware architecture is an increasingly familiar topic in the world of enterprise IT [23]. A Web services are self-contained, modular and distributed environment that can be used across organization boundaries [31]. In the web service as the distributed environment, a service is offered by a provider and is invoked by a requester.

A discovery phase of a web service is one of the most important phases, as the provider advertise its services to the registry and the requester searches registry to find the suitable service [16]. Discovery phase discovers a service based on the functional requirements. Selection phase of a web service address the non-functional properties of the service [24]. Web Service Description Language (WSDL) is used to represent the functional aspects of a Web Service [10]. Web Service Policy (WS-Policy) is used to represent the non-functional properties of a web service [22]. To get the best web service provider for a consumer, both the functional and the non-functional capabilities must be agreed.

Web Service Security Policy (WS-SP) specification is used as a standard for representing security concerns of a web service in different participants. It presents the security concerns as eXtensible Markup Language (XML) tags for each property. In matching security requirements of both provider and consumer, syntactic matching is conducting as it isn't capable of semantic matching [28]. So, there is a need for semantic matching to get more flexible and correct results for matching policies on the two sides of matching.

WS-SP must be presented in Web Ontology Language Description Language (OWL-DL) class [13]. And need to extend this ontology with semantic relations to get the matching level between two security policies. These relations for getting correct results of matching.

In this paper, an improved WS-SP matching algorithm for security concerns in a web service environment is introduced. Through this improvements, simple security policies and complex policies are considered. The best-matched provider from a list of providers is obtained.

This paper focuses on security as one of the important non-functional requirements of a web service. Security of a web service is approved by message security. So, Message security becomes a primary concern when defining Web service. Message security assures how to provide security and protection for SOAP messages that are exchanged in a web service environment [18]. Web service message security provides three main mechanisms: confidentiality, integrity, and availability [17].

A framework is implemented to perform security matching between requirements and capabilities of a web service. In this framework, security classes: message encryption, digital signature, authentication with WS-SP class are associated [17]. Security classes define different techniques and properties of each security mechanism used for securing SOAP message of a web service.

In this paper, we will a semantic matching of WS-SP algorithm improvement. Related work is detailed in section 2. Section 3 presents web service security policy improvements. It details security policy, its problems, and security policy ontology. The complexity of the improved matching algorithm is

presented in section 4. The paper concludes in Section 5.

2. Related work

Security concerns of a distributed SOA environment are important and addressed in industrial and research in the last few years [5, 12]. WS-SP is a framework by W3C and OASIS organizations used for allowing web services to express their security constraints and requirements in the form of policy assertions [29]. In a web service selection phase, researchers suggest negotiation security concerns of each participant for getting agreed provider [30, 32]. But matching is an important step before negotiation.

M. Ben Brahim et al. [4] proposed a semantic technique for specifying and matching the security assertions of a WS-SP. The technique consists on the transformation of WS-SP assertion into an OWL-DL ontology. An algorithm for matching provider and requestor security assertions is proposed. The matching process consists of checking to what extent each security assertion specified in the requester security policy is satisfied by a security assertion specified in the provider security policy. It gets the expected matching degrees between assertions. The matching technique is not detailed.

M. Ben Brahim et al. [6] extends the technique provided in [4]. They presented a semantic technique for specifying and matching web service security policies. They used WS-SP for specifying requester requirements and provider capabilities. Besides to that, the technique presents security requirements and capabilities as OWL ontology. In addition, a semantic reasoner built on top of security ontology to get the matching result. The algorithm performs three main tasks. Firstly, it matches security assertions as presented in [4]. Secondly, it matches security alternatives. Thirdly, security policy matching degree decision is conducted. The overall level of match between them is the highest degree of the match found at the level of any of the requestor alternative and provider alternative pairs. The limitations of this technique are that it compares simple security policies which contain one assertion on each alternative. It doesn't compare complex policies containing more than alternatives which contain a different number of assertions.

Besides to that, comparing security alternatives and policies are not described in details

S. Alhazbi et al. [2] proposed a framework for semantic matching between web service provider and consumer security policies based on preference. It utilizes the alternative feature in WS-SP which allows specifying multi-optional requirements. Ontology used to model the relationships between different web service security concerns. In addition, the reason is used to specify the level of matching. The matching is achieved in two steps: requirement-capability assertions mapping and the final decision (the output). Requirement-capability assertions mapping is conducted by building a matching table that maps requirement assertions to capability ones. The limitations of this technique are that it doesn't consider that a provider may have a capability that a consumer needs. Besides to that, it doesn't consider complex policies.

T-D Cao et al. [9] proposed a semantic technique for defining and matchmaking the web service security policies. This technique transforms WS-SP into the OWL-DL ontology class. It adds semantic relations that can exist between requirements and capabilities of web service sides. It determines the matching level of the provider and requestor security policies. Authors address that matching technique compares simple and complex assertions. A simple assertion is a security assertion which contains only one alternative. Whereas, a complex assertion has more than one alternative. The matching security policy algorithm performs three main tasks as shown in figure 1.

.

Figure 1. Matching security policy algorithm. [9]

The limitations of this technique are that in this technique they mentioned that complex assertions contain more than one alternative and simple assertion contains only one alternative. And according to WS-Policy standards, alternative contains one or more assertion. Besides to that, it lacks processing all probability cases of simple and complex security policies.

The improvement in this paper depends on [4, 6, 9]. It extends WS-SP ontology. Besides, matching technique matches simple and complex policy security policy. Besides, it considers all cases of a simple policy and complex one.

3. Web service security policy matching improvements

In web service, matching the non-functional properties of a web service is important. So, the policy requirements of the requester must be compatible with the capability policies of the provider. WS-SP used to represent the web service security; as a non-functional requirement; concerns for the web services. WS-SP is represented syntactically, so matching compatibility of two policies presents a problem [19]. This paper presents an improvement for getting matched consumer and provider security concerns in a web service environment.

Web service policy consists of different alternatives. Each alternative contains different assertions.
Web service policy contains two operators: "Exactly One" and "Exactly All." "Exactly one" used to express the alternatives that have assertions. Which means that one of the alternatives must hold. Additionally, "Exactly All" means that all children elements must be assured. Assertions describe the requirement options of a requester or a provider that must be held.

An example of WS-SP by a web service as security concerns is that a policy contains one alternative which contains two assertions. A security token presents the signature of the message body must be assured. Besides, it supports the signature of the message body using a Kerberos token [1]. And encryption must be assured using SHA-256 algorithm suite [11].

3.1. Web Service Security Policy and its Problem

WS-SP is represented syntactically, so matching compatibility of two policies presents a problem. Syntactic matching of security policies is a direct process that done by intersection, but it gets semi-correct results and this process lacks semantics [27].
So, security policy standard is not able to get accurate results as different policy uses different vocabularies which may have the same meaning.

In the case of syntactic matching of security requirements and capabilities, it depends on the intersection of policies. Syntactic matching doesn't get effective results of checking requirements of both provider and consumer. As it doesn't take the meaning of the matching process. To illustrate the problem of syntactic matching assumes a person wants to build a Virtual Travel Agency (VTA) that provides services to tourists. This VTA needs to deal with different service providers like hotels or car rental companies or banks to facilitate financial transactions. Security requirements must be restricted in such system. For example, to use a banking service, the system requires that service should encrypt the exchange messages using any one of symmetric algorithms or use RSA algorithm. For authentication, the system states provider to use X.509 or using username/password technique, for the system username/ password is more preferable. The system found two different service providers that satisfy functional requirements, the security properties of these services are depicted in Table 1.

Table 1. Security properties of providers.

Security class	Provider 1	Provider 2
Encryption	AES	RSA
Authentication	X.509	Direct

Automatic syntactical matching will fail with both providers, even though username/password is considered direct authentication. And instead of getting a close match between consumer and provider2, syntactic matching leads to no-match. In order to automatically achieve such matching, a formal model is required to describe the security concepts and their relationships in terms of concepts, subconcepts, and instances. The matching algorithm should find the best map between items in provider's policy and consumer's policy. In the end, the algorithm makes an overall decision based on all requirement-capability items mapping. So, Semantic matching leads to more flexible and correct result of matching policies. Accordingly, matching module is important and needed in a web service environment during selection phase of a web service.

Therefore, it is important to construct a model based on WS-SP that can describe and conclude relations between two security policies. Building such model must add semantics to vocabularies used to build policy and add the classes of message security techniques used to secure policy [25]. This overcomes the deficits of security policy intersection and gets correct results. Semantic ontology is the commonly used concept that used to explicitly specification of this conceptualization [21].

3.2. Adding Semantics to Security Policy

A policy P is defined as a set of policy alternatives $\{Alt_1, Alt_2, \ldots, Alt_N\}$. It is expressed as a disjunction of all its alternatives as follows:

$$P = Alt1 \mid Alt2 \mid \ldots. AltN \quad (1)$$

An alternative Alt is identified as a set of policy assertions $\{Ass_1, Ass_2 \ldots Ass_M\}$. It is also can be expressed as a conjunction of all its assertions as follows:

$$Alt = Ass1 \wedge Ass2 \wedge \ldots. AssM \quad (2)$$

The requested web service security policy *ReqP* can be expressed as follows:

$$ReqP = Alt1 \mid Alt2 \mid \ldots. Alti \quad (3)$$

$$Altj = Ass1 \wedge Ass2 \wedge \ldots. Assj \quad (4)$$

Furthermore, the security policy of a provider *ProvP* can be stated as follows:

$$\text{ProvP} = \text{Alt1} \mid \text{Alt2} \mid \ldots. \text{Altj} \quad (5)$$

$$\text{Altj} = \text{Ass1} \wedge \text{Ass2} \wedge \ldots. \text{Assj} \quad (6)$$

Matching the two security policies; ReqP and ProvP; the target is to find equivalent alternatives as follows:

$$\left((\exists \text{Alt}_i) \text{S.T. } \text{Alt}_i \in \text{ReqP and } (\exists \text{Alt}_j) \text{ S.T. } \text{Alt}_j \in \text{ProvP and } \text{Alt}_i \Leftrightarrow \text{Alt}_j\right) \Rightarrow \left(\text{ReqP} \Leftrightarrow \text{ProvP}\right) \quad (7)$$

To find equivalent alternatives; all assertions in the two policies must be satisfied as expressed in the following rule:

$$\left((\forall \text{Ass}_i) \text{S.T. } \text{Ass}_i \in \text{Alt}_i \text{ and } (\exists \text{Ass}_j) \text{S.T. } \text{Ass}_j \in \text{Alt}_j \text{ and } \text{Ass}_i \Leftrightarrow \text{Ass}_j\right) \Rightarrow \left(\text{Alt}_i \Leftrightarrow \text{Alt}_j\right) \quad (8)$$

The web service security matching technique is detailed in Figure 2. A semantic technique is used for specifying and matching security requirements. Through matching module, WS-SP transformed into the Web Ontology Language Description Language (OWL-DL) [26]. Then, Semantic Web Rule Language (SWRL) used for extending OWL-DL with semantic relations to get the best matching level between the requester and provider policies. These relations lead to correct results for matching security policies.

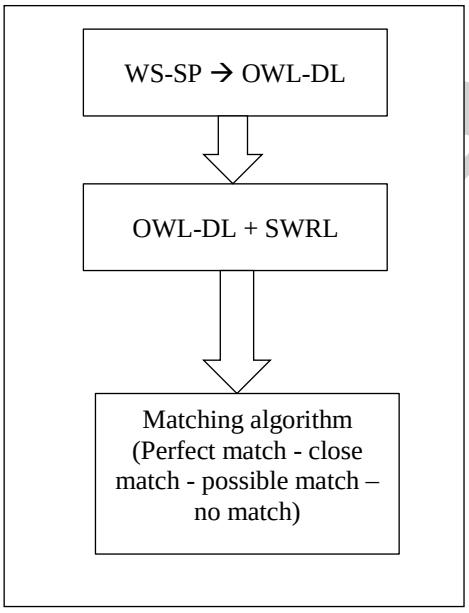

Figure 2. Matching web service security policy.

Transformation of WS-SP into OWL-DL and adding semantic relations to it are presented in related work [7-9]. Semantic relations derived from the ontological representation of WS-SP. Security policy structure and WS-SP concerns must be converted to ontologies in one integrated ontology based model.

To specify SP in ontological representation, there are three classes created "Security Policy" class, "Security Alternative" as a subclass, and "Security Assertion" as a subclass. Security policy comprises one or more alternatives. Security policy alternatives involve of one or more security assertion. Therefore, the three classes created in a particular order where Security Policy is a super class of Security Alternative and Security Alternative is a super class of Security Assertion class.

Besides ontological representation of SP structure, In WS-SP standard, an assertion can have an arbitrary number of types: "Security Binding," "Protection Scope" and, "Supporting Security Tokens." These types reflect the message security of web service requirements or capabilities. Message security must assure the confidentiality, the integrity of data transmitted through the message, and the authentication of the sender [9]. Confidentiality and integrity are guaranteed by applying security mechanisms; encryption and digital signature in sequence. Security tokens used to assure authentication. The ontological representation of security binding assertion types is defined in three classes as below.

The first one, Security Binding class specifies the security mechanism to apply for securing message exchanges [15]. It can be either symmetric binding and asymmetric binding which exemplified by the two subclasses Symmetric Binding and Asymmetric Binding as their types. The second one, Protection Scope class is used to identify message parts of security policy if encrypted or if signed. So, it has two subclasses Encryption Scope and Signature Scope. Signature Scope class contains two subclasses Signed Element and Signed Part [3]. Encryption Scope, according to its definition, has two subclasses Encrypted Element and Encrypted Part. The third one is Supporting Security Tokens class. it creates security binding elements and tokens. It supports tokens that state encryption and signing security requirements. In another word, it guarantees security tokens required by the Security Binding class [14]. It has two classes; Binary Security Token class and XML Security Token class.

In our improvements, we categorize WS-SP as a simple policy and a complex policy. The simple policy is a policy which has zero or one alternative. The complex policy is a policy has more than one alternative. In a policy matching, we categorize matching cases into simple or complex according to a number of alternatives to security policies of consumer and provider. For example, if requester alternatives; expressed as |RAlt|; and provider alternatives; expressed as |PAlt|; equal zero, then it is a simple case. All different cases of simple policy and complex policy are presented in Table 2. From the table, the Simple matching process is conducted in only one case, if requester and provider have one alternative. The complex matching process is carried out if requester and provider have one or more alternatives. There exist

three cases, where a result is obtained directly without conducting and matching steps. So, this decreases execution time of the matching process.

Related works on this topic do not consider all different cases. The improved algorithm studies all possible cases of simple and complex types. After that,

Comparing requester assertions with provider assertions is conducted. There are four possible assertion requirement-capability matching levels: perfect match, close match, possible match and no match. These assertion matching levels depend on semantic relations: "identical," "isLargerThan," "isMoreGeneralThan," "isStrongerThan," "isSmallerThan," etc. it is described in details in [4].

Table 2. Simple policy and complex policy different cases.

Requester	Provider	Case
$\|RALT\| = 1$ or $\|RALT\| > 1$	$\|PALT\| = 0$	**(Simple Policy)** No Match
$\|RALT\| = 0$	$\|PALT\| = 0$	**(Simple Policy)** Perfect Match
$\|RALT\| = 0$	$\|PALT\| = 1$ or $\|PALT\| > 1$	(**Simple Policy**) No Match
$\|RALT\| = 1$	$\|PALT\| = 1$	**Simple Policy** Conduct matching
$\|RALT\| > 1$	$\|PALT\| = 1$	**Complex Policy** Conduct matching
$\|RAlt\| = 1$	$\|PALT\| > 1$	**Complex Policy** Conduct matching
$\|RAlt\| > 1$	$\|PALT\| > 1$	**Complex Policy** Conduct matching

Flowchart for actions of simple policies and complex policies is presented in Figure 3. Matching algorithm mainly depends on a number of alternatives and number of assertions that are the primary components of the security policy. To get the final matching level of simple security policy, get the matching level of assertion as the lowest level of matching found between the requester and provider requirements. In a complex security policy, the matching process matches alternatives of requester security policy with provider alternatives. And in matching alternatives, assertion matching is called. To get the final matching level of a complex security policy, get the matching level as the highest degree of matching found between the requester and provider alternatives. Alternative matchmaker calls assertions matchmaker, which is the simple policy case.

Semantic relations described by Semantic Web Rule Language (SWRL) is added to conclude a relation between policies [20]. These relations bound the requester security policy and provider security policy. These rules define the conditions the requester and provider must satisfy to create a given semantic relations. A sample of SWRL rules are presented below:

- SecurityAssertion(?PAss) ∧ hasOwner(?PAss, "Provider") ∧ SecurityAssertion(?RAss) ∧ hasOwner(?RAss, "Requester") ∧ isStrongerThan(?PAss, ?RAss) → possibleMatch(?RAss, ?PAss).
- SecurityAlternative(?PAlt) ∧ hasAssertion(?PAlt, ?PAss) ∧ SecurityAlternative(?RAlt) ∧ hasAssertion(?RAlt, ?RAss) ∧ PossibleMatch(?PAss, ?RAss) → PossibleMatch(?PAlt, ?RAlt).
- SecurityAssertion(?PAss) ∧ SecurityAssertion(?RAss) ∧ hasManyCloseRelationsWith(?PAss, ?RAss) → CloseMatch(?PAss, ?RAss).
- SecurityAssertion(?PAss) ∧ SecurityAssertion(?RAss) ∧ isSmallerThan(?PAss, ?RAss) → NoMatch(?PAss, ?RAss).
- AlgorithmSuite(?PAlgS) ∧ AlgorithmSuite(?RAlgS) ∧ hasAnalogAlgS(?PAlgS, ?RAlgS) ∧ hasEncryption(?PAlgS, ?PEnc) ∧ hasValue(?PEnc, "Aes192") ∧ hasEncryption(?RAlgS, ?REnc) ∧ hasValue(?REnc, "Aes128 ") → isStrongerThan(?PEnc, ?REnc).

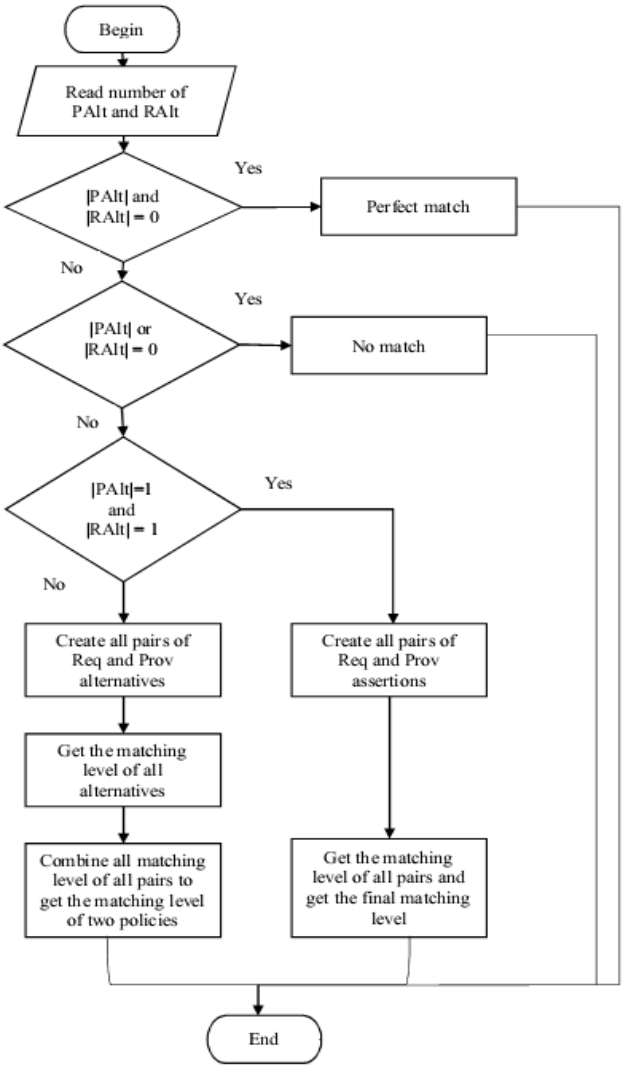

Figure 3. Simple policy and complex policy matching.

4. Complexity of the Improved Matching Algorithm

The complexity of matching module that matches p olicies is analyzed by including the elements in Requester Policy *ReqP* with a number of alternatives and Provider Policy *ProvP*.

Table 3 defines a complexity for the improved matching algorithm compared to previous work [4] and [9]. For a simple security policy, Complexity of the improved matching algorithm is the same as in [4, 9], which takes into account only the number of assertions of requester and provider. For simple policy cases, if any of participants alternatives equal "0", then complexity equal "0". If a number of alternatives in each participant equal "1", then complexity equal to the number of assertions in each participant. If a number of requester alternatives greater than or equal to "1" and provider alternatives equal "1", then complexity equal to the number of provider alternatives multiplied by the number of assertions of provider and number of requester assertions. All other cases complexity computed with the same way. Note that, the complexity of complex policy is defined in the improved matching algorithm only. As the improved matching algorithm considers the complex policy cases. The gray cells represent that this case is not considered in this work.

Table 3. Complexity of improved matching algorithm.

Requester	**Provider**	**Work [4]**	**Work [9]**	**Improved matching algorithm**
|RAlt|=1 or |RAlt| > 1	*|PAlt| = 0*			*0*
|RAlt| = 0	*|PAlt| = 0*			*0*
|RAlt| = 0	*|PAlt| =1 or |PAlt| > 1*			*0*
|RAlt| = 1	*|PAlt| = 1*	*O(|RAss|.|PAss|)*	*O(|RAss|.|PAss|)*	*O(|RAss|.|PAss|)*
|RAlt| >1	*|PAlt| = 1*			*O(|RAlt|.|RAss|.|PAss|)*
|RAlt| = 1	*|PAlt| > 1*			*O(|PAlt|. |RAss|.|PAss|)*
|RAlt| > 1	*|PAlt| > 1*			*O(|RAlt|.|PAlt|. |RAss|.|PAss|)*

As a generalization of matching web service security policy, the consumer matches its requirements with *N* providers to get the best suitable provider. This will makes execution complexity to be N multiplied by the big O notation of case of matching.
A parallel technique can be used to decrease the processing time of a matching WS-SP of N providers.
In the parallel technique , processing of matching the requester with each provider executed separately.

5. Conclusion and Future Directions

In this paper, improved web service security policy-matching algorithm is introduced with considering all cases of simple security policy and complex security policy. In addition, it states all simple and complex policy cases during matching of web service security policy matching. SWRL rules are added to get the relation between security policies represented as ontology.A generalization of the matching algorithm is introduced to get the best-matched provider from a list of providers. a parallel technique can be used with the generalized matching process to decrease processing time.

If the matching technique is not enough to get a matched requirement for both participants, Negotiation is the direction to try to get an agreement between requirements of both provider and consumer security concerns.

As a future work, we aim to include a matching process in the discovery phase of a web service to address the effect of the real including of a matching process in a discovery phase of a web service. Also, we intend to extend the matching algorithm with a negotiation technique if a matching failed to get a result. Besides, we target to address where interaction between web service provider and the consumer can take place. In addition, we intend to apply matching and negotiation in a real world environment.

References

[1] Adams C. and Pinkas D., "Internet X. 509 public key infrastructure time-stamp protocol (TSP)," 2001.

[2] Alhazbi S., Khan K. M., and Erradi A., "Preference-based semantic matching of web service security policies," in *2013 World Congress on Computer and Information Technology (WCCIT)*, 2013.

[3] Barhoom T. S. and Rasheed R. S., "Position of signed element for SOAP message integrity," *International Journal of Computer Information Systems*, vol. 2, pp. 21-28, 2011.

[4] Ben Brahim M., Chaari T., Jemaa M. Ben, and Jmaiel M., "Semantic matching of web services security policies," in *Risk and Security of Internet and Systems (CRiSIS), 2012 7th International Conference on*, 2012, pp. 1-8.

[5] Bertino E., Martino L., Paci F., and Squicciarini A., *Security for Web services and service-oriented architectures*: Springer Science & Business Media, 2009.

[6] Brahim M. B., Chaari T., Jemaa M. B., and Jmaiel M., "Semantic matching of ws-securitypolicy assertions," in *Service-Oriented Computing-ICSOC 2011 Workshops*, 2012, pp. 114-130.

[7] Brahim M. B., Chaari T., Jemaa M. B., and Jmaiel M., "Semantic Matching of WS-SecurityPolicy Assertions," in *ICSOC Workshops*, 2011, pp. 114-130.

[8] [8] Brahim M. B., Chaari T., Jemaa M. B., and Jmaiel M., "Semantic matching of web services security policies," in *Risk and Security of Internet and Systems (CRiSIS), 2012 7th International Conference on*, 2012, pp. 1-8.

[9] Cao T.-D. and Tran N.-B., "Enhance Matching Web Service Security Policies with Semantic," in *Knowledge and Systems Engineering*, ed: Springer, 2014, pp. 213-224.

[10] Chinnici R., Moreau J.-J., Ryman A., and Weerawarana S., "Web services description language (wsdl) version 2.0 part 1: Core language," *W3C recommendation*, vol. 26, p. 19, 2007.

[11] Chung H. V., Nakamura Y., and Satoh F., "Security Policy Validation For Web Services," ed: Google Patents, 2007.

[12] Dawoud W., Takouna I., and Meinel C., "Infrastructure as a service security: Challenges and solutions," in *Informatics and Systems (INFOS), 2010 the 7th International Conference on*, 2010, pp. 1-8.

[13] De Bruijn J., Lara R., Polleres A., and Fensel D., "OWL DL vs. OWL flight: conceptual modeling and reasoning for the semantic Web," in *Proceedings of the 14th international conference on World Wide Web*, 2005, pp. 623-632.

[14] Garcia D. Z. G. and Toledo M. B. F. De, "Ontology-based security policies for supporting the management of web service business processes," in *Semantic Computing, 2008 IEEE International Conference on*, 2008, pp. 331-338.

[15] Gruschka N. and Iacono L. L., "Vulnerable cloud: Soap message security validation revisited," in *Web Services, 2009. ICWS 2009. IEEE International Conference on*, 2009, pp. 625-631.

[16] [16] Guinard D., V. Trifa, Karnouskos S., Spiess P., and Savio D., "Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services," *Services Computing, IEEE Transactions on*, vol. 3, pp. 223-235, 2010.

[17] Jamil D. and Zaki H., "Security issues in cloud computing and countermeasures," *International Journal of Engineering Science and Technology (IJEST)*, vol. 3, pp. 2672-2676, 2011.

[18] Jensen M., Gruschka N., and Herkenhöner R., "A survey of attacks on web services," *Computer Science-Research and Development*, vol. 24, pp. 185-197, 2009.

[19] Kagal L., Finin T., and Joshi A., "A policy based approach to security for the semantic web," in *International Semantic Web Conference*, 2003, pp. 402-418.

[20] Liu C.-H., Chang K.-L., Chen J. J.-Y., and Hung S.-C., "Ontology-based context representation and reasoning using owl and swrl," in *Communication Networks and Services Research Conference (CNSR), 2010 Eighth Annual*, 2010, pp. 215-220.

[21] Martin D., Paolucci M., McIlraith S., Burstein M., McDermott D., McGuinness D., *et al.*, "Bringing semantics to web services: The OWL-S approach," in *Semantic Web Services and Web Process Composition*, ed: Springer, 2005, pp. 26-42.

[22] Rosenberg J. B. and Remy D. L., *Securing Web Services with WS-Security: Demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption*: Sams, 2004.

[23] Sahni Y., Cao J., and Liu X., "MidSHM: A Middleware for WSN-based SHM Application using Service-Oriented Architecture," *Future Generation Computer Systems*, 2017.

[24] Sahoo B. and Bhuyan P., "A selection approach in service composition of SOA," in *Recent Trends in Information Technology (ICRTIT), 2016 International Conference on*, 2016, pp. 1-6.

[25] Serme G., Oliveira A. S. de, Massiera J., and Roudier Y., "Enabling message security for RESTful services," in *Web Services (ICWS), 2012 IEEE 19th International Conference on*, 2012, pp. 114-121.

[26] Sirin E., Parsia B., Grau B. C., Kalyanpur A., and Katz Y., "Pellet: A practical owl-dl reasoner," *Web Semantics: science, services and agents on the World Wide Web*, vol. 5, pp. 51-53, 2007.

[27] Speiser S., "Semantic annotations for ws-policy," in *Web Services (ICWS), 2010 IEEE International Conference on*, 2010, pp. 449-456.

[28] Vedamuthu A. S., Orchard D., Hirsch F., Hondo M., Yendluri P., Boubez T., *et al.*, "Web services policy 1.5-framework," *W3C Recommendation*, vol. 4, pp. 1-41, 2007.

[29] [Vedamuthu A. S., Orchard D., Hirsch F., Hondo M., Yendluri P., Boubez T., *et al.*, "Web services

policy 1.5-framework," *W3C Working Draft*, 2006.
[30] Zheng X., Martin P., Powley W., and Brohman K., "Applying bargaining game theory to web services negotiation," in *Services Computing (SCC), 2010 IEEE International Conference on*, 2010, pp. 218-225.
[31] Zheng Z., Zhang Y., and Lyu M. R., "Distributed qos evaluation for real-world web services," in *Web Services (ICWS), 2010 IEEE International Conference on*, 2010, pp. 83-90.
[32] Zulkernine F. H. and Martin P., "An adaptive and intelligent SLA negotiation system for web services," *IEEE Transactions on Services Computing*, vol. 4, pp. 31-43, 2011.

Arabi keshk received the B.Sc. in Electronic Engineering and M.Sc. in Computer Science and Engineering from Menoufia University, Faculty of Electronic Engineering in 1987 and 1995, respectively and received his Ph.D. in Electronic Engineering from Osaka University, Japan in 2001. His research interest includes software testing, software engineering, distributed system, database, data mining, and bioinformatics.

Ashraf El-Sisi received the B.Sc. and M.Sc. in Electronic Engineering and Computer Science Engineering from Menofia University, Faculty of Electronic in 1989 and 1995, respectively and received his Ph.D. in Computer Engineering & Control from Zagazig University, Faculty of Engineering in 2001. His research interest includes cloud computing, privacy preserving data mining, and Intelligent systems.

Mohamed Elkawkagy, (1973) ،male, Faculty of Computers and Information, Menoufia University, Egypt, Lecturer, received his Ph.D. in 2012, his research directions include AI-planning, Software Engineering, Planning search strategy ،Multi-agent Planning, Web-based planning, Agent Systems and Human Computer Interaction (HCI)

Amira Abdelatey received the B.Sc., M.Sc., and PhD. in computers and information from Menoufia University, Faculty of computers and information in 2007, 2012, and 2017 respectively.
Currently Lecturer at computer Science department at Faculty of computers and information, Menoufia University.
Her research interest includes semantic web, web service, intelligent systems, web service security, database system and software engineering.