Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management
Istanbul, Turkey, July 3 – 6, 2012

Study of Access Control Issue for Web Services

Hadiseh Seyyed Alipour
Department of Software Engineering
Qazvin Islamic Azad University, Qazvin 1416-34185, Iran

Mehdi Sabbari
Department of Software Engineering
Qazvin Islamic Azad University, Qazvin 1416-34185, Iran

Abstract

Security is an important issue that must be well-defined in Service Oriented Architecture (SOA) environment, so that it could be used in implementing the web services. In this article, we focus on one of the important aspects of SOA security, which is access control. The article explains the security requirements that must be followed and proposes a conceptual model of requirements in this field based on the needs. Then every requirement, available techniques and standards in this filed is separated and discussed. Since different models such as IBAC, RBAC, ABAC and RAdAC have been presented so far, these existing models are explained. Then a comparison between ABAC model's structure that is more compatible with SOA and RBAC model that is most widely used today is presented.

Keywords
Service Oriented Architecture; Access Control; Security Requirements; RBAC; ABAC.

1. Introduction
Service-oriented computing represents a new generation distributed computing platform. As such, it encompasses many things, including its own design paradigm and design principles, design pattern catalogs, pattern languages, a distinct architectural model, and related concepts, technologies, and frameworks (Erl 2007). Web services seem to become the preferred implementation technology for realizing the SOA promise of maximum service sharing, reuse, and interoperability (Parveen and Tilley 2008). Web Service supports the communication between applications developed on different platform by different programming languages with different technological standards. The ultimate goal of Web Service is to realize the integration and interaction between various systems in Internet/Intranet environment, just similar to the function of components (Wang et al. 2009). Web services technology platform is comprised of the following core open technologies and specifications: Web Services Description Language (WSDL), XML Schema Definition Language (XSD), SOAP (formerly the Simple Object Access Protocol), UDDI (Universal Description, Discovery, and Integration), and the WS-I Basic Profile (Karp 2006). Access control security is one of the important aspects in SOA that is considered as a challenge. This issue requires further attention and review because of the architecture's distributed nature, its high re-usability, simple accessibility and the autonomy of logical solutions units. A number of models such as (IBAC, RBAC, ABAC, RAdAC) have been developed to address various aspects of access control problem.

In the section 2 of the paper, the security Requirements that must be followed in the field of access control are explained and a conceptual model with a collection of Requirements is proposed. Then in the section 3, there is a comprehensive classification of all available technologies that meet the security Requirements; all of them will be explained. Section 4 belongs to the description of IBAC, RBAC, ABAC and RAdAC models, here the ABAC model's benefits and structure is explained; the ABAC model is more compatible with SOA. ABAC model's descriptions works better in the section 5 of the article, when we present a comparison between ABAC and RBAC models using an example. In the section 6 the new challenges and future work is pointed out. Finally we conclude the paper in section 7.

2. Security Requirements in Access Control

Security is an important issue that must be well-defined in SOA environment so that it could be used in implementing the web services. In general and according to (Hafner and Breu 2009), security in the environment can be defined in this way: "the sum of all techniques, methods, procedures and activities employed to maintain an ideal state specified through a set of rules of what is authorized and what is not in a heterogeneous, decentralized, and inter-connected computing system". Security objectives provide a categorization of the most basic security needs of an asset. Define in (Hafner and Breu 2009): "a statement of intent to counter identified threats and/or satisfy identified organization security policies and assumptions". Based on our research in the field of SOA security, we've reached the conclusion that the SOA environment should be divided into (transport & network, message, interaction between services, services policies, access control and applications) domains, and Security should be respected in any field. Therefore we did polling among thirty Experts in computer and information technology, and we obtained useful results. One goal of conducted poll is determining the importance of each security principles in every area. Accordingly as shown by figure1, in the field of access control, the most important needs are authentication, authorization and accountability needs, possess availability and federation, the next rank.

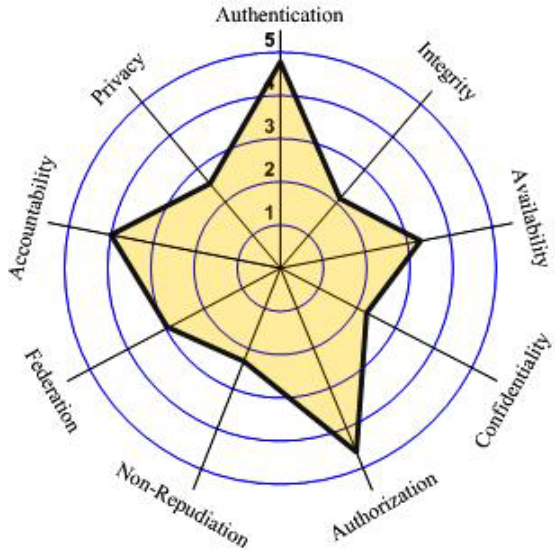

Figure 1: Spider diagram of polling results in the rate of the importance of each security principles in access control

Based on our proposed conceptual model as shown by Figure 2, the total security requirements in the area of access control are divided into two general categories: functional aspects and non-functional aspects that the non-functional aspects of Security are within the functional aspects of security and cover them. Functional aspects in the area include (Authentication, Authorization, Accountability and Privacy) and must seriously observance for more security in this area. Non-functional aspects include (Auditing & Compliance, Interoperability, Manageability and Ease of Development). Finally, both practical and impractical aspects of security in access control stand within the domain of SOA rule and training, awareness, and risk management.

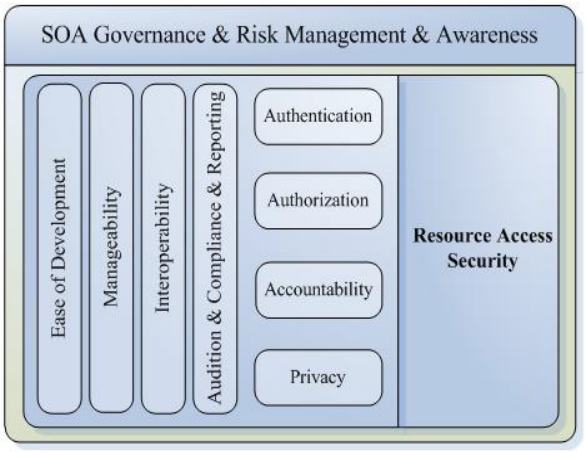

Figure 2: Security requirements conceptual model in access control

3. Available Techniques in Access Control Filed

Several organizations, including OASIS, W3C, the Liberty Alliance, and various members of industry have put together numerous security standards and techniques for securing Web services. For the most part, these standards and techniques all complement or extend one another, but there are some conflicting or competing standards (Singhal and et al. 2007). This section provides an overview of the various standards and how they can be used to meet security requirements and protect against threats to Web services. These standards are shown in Table 1. WS-Security (Web Service Security) specifies SOAP security extensions that provide confidentiality using XML Encryption and data integrity using XML Signature (Singhal and et al. 2007, Crampton and et al. 2007). WS-Security also includes profiles that specify how to insert different types of binary and XML security tokens in WS-Security headers for authentication and authorization purposes (Chanliau 2006):

- Username with optional Password digest (defines how a web service consumer can supply a username as a credential for authentication.
- X.509 Certificate (a signed data structure designed to send a public key to a receiving party).
- Kerberos ticket (an authentication and session token).
- REL document (rights expression language (REL) license tokens inserted in WS-Security headers are used for authorization).
- XCBF document (defines how to use the XML Common Biometric Format (XCBF) language for authentication with the WS-Security specification).

EPAL (Enterprise Privacy Authorization Language) is a formal language for writing enterprise privacy policies to govern data handling practices in IT systems according to fine-grained positive and negative authorization rights. Logging operation for recording the events and accidents meet the accountability needs (Janssen 2008). SAML (Security Assertion Markup Language) defines an XML vocabulary for sharing security assertions that specify whether and how an entity was authenticated, information about an entity's attributes or whether an entity is authorized to perform a particular action. These assertions enable identity federation and distributed authorization within a SOA (Singhal and et al. 2007). SAML is an open framework for sharing security information on the Internet through XML documents. SAML was originally designed to address the following (Chanliau 2006):

- Limitations of web browser cookies: SAML provides a standard way to transfer cookies across multiple Internet domains.
- Proprietary web single sign-on (SSO): SAML provides a standard protocol to implement SSO within a single domain or across multiple domains.
- Federation: SAML enables identity management (a user can have several identities on the Internet).
- Web services security: SAML provides a standard security token (a SAML assertion) that can be used with the WS-Security framework.

Table 1: Requirement and standards for access control

Area	Requirements	Standards
Access Control	Authentication	X.509 Certificate
		Kerberos ticket
		WS-Security Token
		REL
		XCBF
	Authorization	IBAC/RBAC / ABAC /RAdAC
		SAML
		XACML
	Privacy	EPAL
	Accountability	Logging

XACML (eXtensible Access Control Markup Language) is an OASIS standard that describes both a policy language implemented in XML and an access control decision request/response language implemented in XML (Singhal and et al. 2007). The policy language details general access control requirements, and has standard extension points for defining new functions, data types, combining logic, etc. The request/response language lets you form a query to ask whether or not a given action should be allowed, and interpret the result. The response always includes an answer about whether the request should be allowed using one of four values: Permit, Deny, Indeterminate (an error occurred or some required value was missing, so a decision cannot be made) or Not Applicable (the request can't be answered by this service) (Moses and et al. 2005).

4. Comparison of available approaches in access control

This section describe the authorization models most relevant to access management in a SOA, namely identity-based, role-based, attribute-based, and risk-adaptive access control.

4.1 Identity Based Access Control

Under this model, permissions to access a resource is directly associated with a subject's identifier (e.g., a user name). Access to the resource is only granted when such an association exists. An example of IBAC is the use of Access Control Lists (ACL), commonly found in operation systems and network security services (Yuan and Tong 2005). The concept of an ACL is very simple: each resource on a system to which access should be controlled, referred to as an object, has its own associated list of mappings between the set of entities requesting access to the resource and the set of actions that each entity can take on the resource. Drawbacks of IBAC are: the number of identifiers in the ACL will increase and become difficult to maintain as more users request access, making this approach impossible to scale. The ACL for a particular file, process, or other resource must be checked every time the resource is accessed, and this can be an inefficient means of providing access control. Access control decisions are not based on any business function or characteristics of the user but solely on the identifiers, making it unsuitable for enterprise level use.

4.2 Role Based Access Control

The RBAC model restricts access to a resource based on the business function or role the subject is performing. The permissions to access a resource are then assigned to the appropriate role(s), rather than directly assigned to subject identifiers (Sandhu and et al. 1996). When a user changes jobs, some other user is allowed to take on that role. No ACL changes are needed. Of course, sometimes only a few of the user's rights change. In that case, a new role needs to be introduced. Often the rights associated with a role depend on which user is acting in that role. In that case, too, a new role needs to be introduced (Ferraiolo and Kuhn 1992). The RBAC reference model is defined in terms of four model components: Core RBAC, Hierarchical RBAC, Static Separation of Duty Relations, and Dynamic Separation of Duty Relations (Kuhn 2003). Although RBAC may take slightly different forms, a common representation as defined in (WU and XI 2009) is depicted in Figure 3.

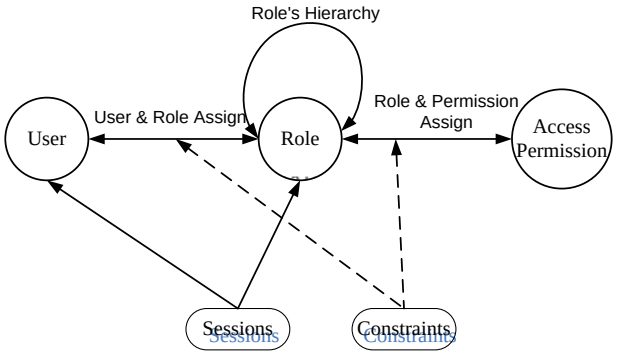

Figure 3: Role-based access control model

In most cases, Web service platforms will support designation and assignment of privileges to roles as part of their standard definition of user accounts and access control privileges. In worst cases, the administrator will have to create the necessary user groups, enroll the appropriate users, and assign them role-appropriate privileges. RBAC on a Web service platform should be implemented at a minimum for the administrator, developers, and any other privileged accounts that will be required for the Web service to operate. The Web service platform must be configured to enforce separation of roles (i.e., not allowing a user assigned to one role to perform functions exclusively assigned to another role). The privileges associated with each role should be assigned in a way that implements least privilege each role should be assigned only the minimum privileges needed to perform the functions required by the role (Rolls 2008). Some RBAC limitations in SOA are: In RBAC you still have to manage every user account and bind these accounts to roles. Unknown accounts can only be linked to a default role, like guest or customer. Because the core RBAC model limits the abstraction of user function to roles only, it does not consider any other characteristics that a user may demonstrate. Further, RBAC generally doesn't take into account the characteristics of resources (other than their identifiers); nor does it capture any security relevant information of the environment.

4.3 Attribute Based Access Control

Policy-Based Access Control (PBAC), which is called Attribute-Based Access Control (ABAC) in the US Defense Department jargon, extends RBAC to a more general set of properties (Karp and Li 2010). Unlike IBAC and RBAC, the ABAC model (Yuan and Tong 2005) can define permissions based on just about any security relevant characteristics, known as attributes. For access control purposes, we are concerned with three types of attributes:

- Subject Attributes (S). Associated with a subject that defines the identity and characteristics of that subject.
- Resource Attributes (R). Associated with a resource, such as a Web service, system function, or data
- Environment Attributes (E). Describes the operational, technical, or situational environment or context in which the information access occurs.

In the most general form, a Policy Rule that decides on whether a subject s can access a resource r in a particular environment e, is a Boolean function of s, r, and e's attributes:

(1) $$Rule\ X : can\_access(s, r, e) \leftarrow f(ATTR(s), ATTR(r), ATTR(e)).$$

ABAC clearly provides an advantage over traditional RBAC when extended into SOA environments, which can be extremely dynamic in nature. ABAC policy rules can be custom-defined with consideration for semantic context and are significantly more flexible than RBAC for fine-grained alterations or adjustments to a subject's access profile. ABAC also integrates seamlessly with XACML, which relies on policy-defined attributes to make access control decisions. One additional benefit to Web service implementations of ABAC lies in the nature of the loose definition of subjects. Because ABAC provides the flexibility to associate policy rules to any actor, it can be extended to Web service software agents as well (Tong 2005).

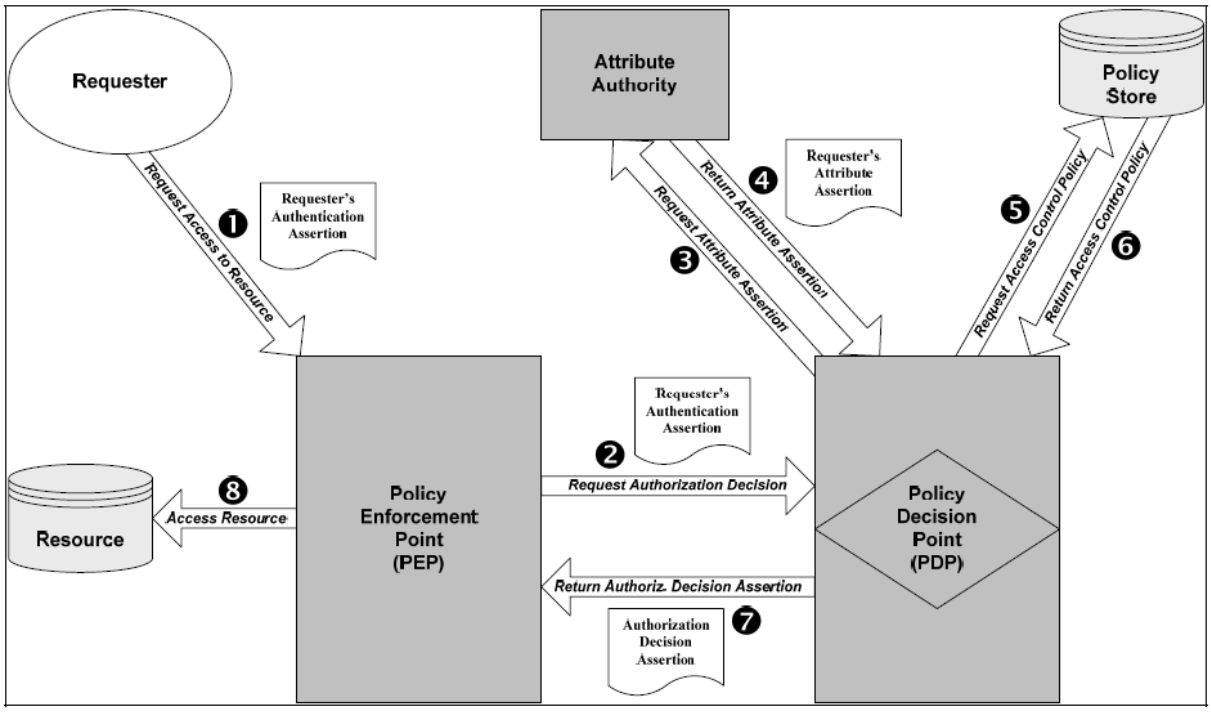

Figure 4: Use of SAML and XACML in implementing ABAC

One additional benefit to Web service implementations of ABAC lies in the nature of the loose definition of subjects. Because ABAC provides the flexibility to associate policy rules to any actor, it can be extended to Web service software agents as well. Figure 4 illustrates how an ABAC attribute authority (AA) can be integrated with a SAML framework. In this diagram, the AA generates attribute assertions, which contain all the attributes necessary for an access control decision based on an ABAC policy written in XACML. The PDP uses the attribute assertions, the authentication assertion, and the XACML policy to generate an authorization decision assertion (Singhal and et al. 2007).

In Figure 4, the requester's authentication assertion is provided by the identity provider before accessing the resource. The following steps describe how SAML and XACML use the requester's attributes to determine whether access should be granted: (1) the requester attempts to access the resource and supply the authentication assertion. (2) The Policy Enforcement Point (PEP) sends a SAML authorization decision request to the PDP. (3) The PDP requests certain attribute assertions that are associated with the requester. (4) The AA returns the appropriate attribute assertions. (5) The PDP requests the XACML policy from the policy store. (6) The PDP receives the XACML policy. (7) After querying the XACML policy, the PDP sends an authorization decision assertion to the PEP. (8) Based on the authorization decision assertion, the PEP grants the requester access to the resource.

4.4 Risk Adaptive Access Control

Risk adaptive access control (RAdAC) (Cheng and et al. 2007) is another variation access control methods. As opposed to IBAC, RBAC and ABAC, however, RAdAC makes access control decisions on the basis of a relative risk profile of the subject and not necessarily strictly on the basis of a predefined policy rule. Figure 5, illustrates the logical process governing RAdAC, which uses a combination of a measured level of risk the subject poses and an assessment of operational need as the primary attributes by which the subject's access rights are determined.As a policy-driven mechanism, RAdAC is ostensibly an abstraction of ABAC. Unlike ABAC, however, a RAdAC framework requires associations with sources that are able to provide real-time, situation aware information upon which risk can be assessed with each authentication request.

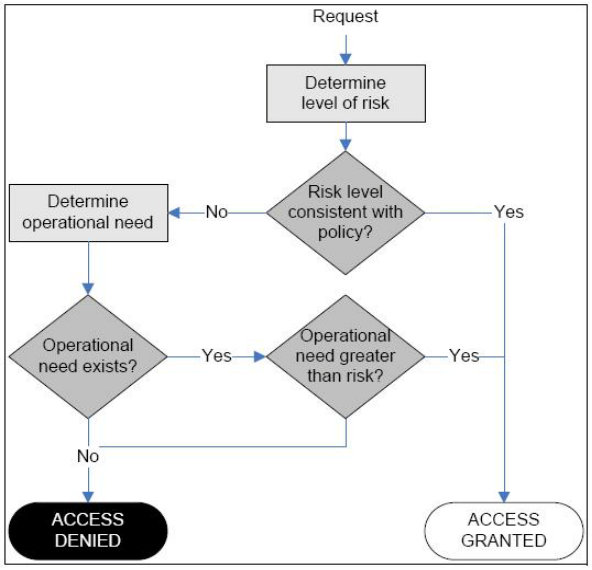

Figure 5: RAdAC decision tree

5. Compare ABAC VS. RBAC with example

Since many access control solutions today are role based, in this section we compare ABAC with the latest RBAC approaches to illustrate the advantages of this new model. We use an example that involves a slightly more complex access control scenario to show how RBAC and ABAC attack the problem differently (Yuan and Tong 2005). In example, an Online Entertainment Store streams movies to users for a flat monthly fee. The store needs to enforce an access control policy that is based on the users' age and the movie content ratings. The movie ratings and the corresponding access control policy is listed in Table 2:

Table 2: Movie rating and user allowed

Movie Rating	**User Allowed**
R	Age 21 or older
PG-13	Age 13 or older
G	Everyone

In the standard RBAC model, where only user roles are considered, there would be three pre-defined roles created for users, Adult, Juvenile, and Child. Each user of the store would be assigned to one of the three roles, possibly during registration. There would be three permissions created, namely, Can view R rated movies, Can view PG-13 rated movies and can view G rated movies, each represent the permission to view movies with the respective rating. The Adult role gets assigned with all three permissions, whereas the Juvenile role gets Can view PG-13 rated movies and Can view G rated movies permissions, and the Child role gets the Can view G rated movies permission only. Both the user-to-role assignment and the permission-to-role assignment are manual administrative tasks. In comparison, the ABAC approach in this scenario has no need to explicitly define roles. Instead, whether a user u can access or view a movie m (in a security environment e which is ignored here) would be resolved by evaluating a policy rule such as the following:

$R1: can\_access(u, m, e) \leftarrow$
$(Age(u) \geq 21 \wedge Rating(m) \in \{R, PG-13, G\}) \vee$
$(21 > Age(u) \geq 13 \wedge Rating(m) \in \{PG-13, G\}) \vee$
$(Age(u) < 13 \wedge Rating(m) \in \{G\})$

Where Age and Rating are the subject attribute and the resource attribute, respectively. The advantage of the ABAC model shown here is that it eliminates the definition and management of static roles, hence also eliminates the need for the administrative tasks for user-to-role assignment and permission-to-role assignment. Finer-grained access

control policies often involve multiple subject and object attributes. In such cases ABAC becomes more manageable and scalable than RBAC. To illustrate this, we extend the example slightly: suppose movies are also categorized either as *New Release* or as *Old Release* based on release dates, and users are further classified as *PremiumUser* and *Regular User* based on the membership fee paid. Suppose the store would like to enforce a policy such that only premium users can view new releases. In RBAC, the roles and permissions that need to be created have both doubled:

Table 3: Roles and permissions in RBAC

Standard RBAC Roles
Adult premium user role
Adult regular user role
Juvenile premium user role
Juvenile regular user role
Child premium user role
Child regular user role

Standard RBAC Permissions
Can view R rated new release
Can view R rated old release
Can view PG-13 rated new release
Can view PG-13 rated old release
Can view G rated new release
Can view G rated old release

In general, if there are K subject attributes and M resource attributes, and if for each attribute, Range() denotes the range of possible values it can take, then the respective number of roles and permissions need to be created would be:

(2) $\prod_1^k Range(SA_K)$

and

(3) $\prod_1^M Range(RA_m)$

Therefore, as the policy becomes finer-grained and more attributes are involved, the number of roles and permissions will grow exponentially, making the user-to-role assignment and permission-to-role assignment tasks prohibitively expensive.
In the ABAC model, the previous policy rule R3 still applies; we only need to add a second rule R4, and then combine the two conjunctively:

$$R_4: can\_access(u, m, e) \leftarrow$$
$$(MembershipType(u) =' Premium')$$
$$\vee (MembershipType(u) =' Regular' \wedge MovieType(m) = 'OldRelease')$$

$$R5: can\_access(u, m, e) \leftarrow R3 \wedge R4$$

So far we haven't addressed environment attributes. The RBAC model does not address environment attributes explicitly. For example, it would be even more awkward to implement a policy that states "*Regular users are allowed to view new releases in promotional periods*." By contrast, in an ABAC model, this policy can be

implemented simply by adding conjunctive sub-clauses to check to see if today's date, an environment attribute, falls in a promotional period.

6. New Challenges and Future Works

As mentioned above ABAC helps us overcome a number of problems that previous access control models could not solve, and some of the issues previous models created for us. But new challenges that ABAC adds are:

- The quality of privilege-giving attributes becomes crucial. So attributes need to be maintained in a secure manner. Thus, in the same way we need governance in maintenance of identities today, ABAC may add a need to introduce governance in attribute maintenance.
- ABAC lets us shift to an abstraction layer of policies and rules which attributes are used to enable efficient rule evaluations. These rules can be made precise and fine-grained. But which attributes we use and how we use them is in no way standardized and how we capture and write sufficient rules are not defined, for example the failure to capture a business requirement for Separation of Duty (SoD) would be the result of imprecise or erroneous rule definitions.
- ABAC introduces a dynamic approach to access control which is good. But auditing needs to be approached in a new fashion. Who gets access to what at any point in history is determined by a combination of a (the policies in effect and b) the attributes assigned to users and resources. So auditors will need tools to analyze changes made both to attributes and policies.

So, for future work we attempt to present a model that overcomes some of these issues.

7. Conclusion

The article's focus is on reviewing the security in the field of access control in the SOA environment. To obtain this purpose, security requirements, elements and approaches followed by proposing a conceptual model. The available approaches and standards in the field were explained. Further, a comparison between the available models in access control fields was discussed followed by ABAC model's benefits, and it is understood that ABAC model is more compatible with SOA environment.

References

Chanliau, M., Web Services Security: What's Required To Secure A Service-Oriented Architecture, *An Oracle White Paper*, 2006.

Cheng, P. C., Rohatgi, P. and Keser, C., Fuzzy MLS: an experiment on quantified risk–adaptive access control, In *2007 Proc. IEEE Symposium on Security and Privacy*, pp. 222-230, 2007.

Crampton, J., Wei Lim, H., and G.Paterson, K., What Can Identity-Based Cryptography Offer to Web Services?, *ACM*, Virginia, USA, 2007.

Eckert, J., Bachhuber, M., Miede, A., Pasageorgiou, A., and Steinmetz, R., Service-oriented Architectures in the German Banking Industry-A Multi-Participant Case Study, *4th IEEE International Conference on Digital Ecosystems and Technologies* (IEEE DEST 2010), 2010.

Erl, T., SOA: Principles of Service Design, Prentice Hall/Pearson PTR, 2007.

Ferraiolo, D.F., and Kuhn, D.R., Role Based Access Control, *15th National Computer Security Conf.* pp. 554-563, 1992.

Fiere, J., SOA Security, *Master Thesis*, Faculty of Science Vrije Universiteit Amsterdam, 2007.

Hafner, M., and Breu, R., Security Engineering for Service-Oriented Architectures, *Springer*, 2009.

Jana, D., Chaudhuri, A., and Bhaumik, B., Privacy and Anonymity Protection in Computational Grid Services, *International Journal of Computer Science and Applications*, Vol, 6, No, 1, pp. 98-107, 2009.

Janssen, J., Identity management within an organization, *Master Thesis*, Radbound University Nijmegen, 2008.

Jonnaganti, V., An Integrated Security Model for the Management of SOA- Improving the attractiveness of SOA Environments through a strong Architectural Integrity, *Master Thesis*, University of Gothenburg Department of Applied Information Technology Gothenburg, Sweden, 2009.

Kanneganti, R., and Chodavarapu, P. A., SOA Security, *Manning*, 2008.

Karp, H., A., Authorization-Based Access Control for the Services Oriented Architecture, in the *Fourth International Conference on Creating, Connecting, and Collaborating through Computing, IEEE*, Berkeley, CA, USA, 2006.

Karp, A. H. and Li, J., Solving the Transitive Access Problem for the Services Oriented Architecture, *IEEE International Conference on Availability, Reliability and Security*, DOI 10.1109/ARES, 2010.

Kuhn, R., Role Based Access Control, *American National Standards Institute*, 2003.

Moses T., and et al, eXtensible Access Control Markup Language (XACML) Version 2.0, *OASIS Standard*, 2005.

Papazoglou, M. P., and Van Den Heuvel, W., Service oriented architectures: approaches, technologies and research issues, pp. 389-415, *Springer-Verlag*, 2007.

Parveen, T., and Tilley, S., A Research Agenda for Testing SOA-Based Systems, *SysCon 2008-IEEE International Systems Conference*, Montreal, Canada, 2008.

Rolls, D., Establishing an operational context for shared role-based access control systems, *White Paper, SailPoint Technologies*, 2008.

Sandhu, R., Coyne, E.J., Feinstein, H.L., and Youman, C.E. Role-Based Access Control Models. *IEEE Computer* 29(2), pp. 38-47, 1996.

Singhal, A., Winograd, T., and Scarfone, K., Guide to Secure Web Services, *National Institute of Standards and Technology Special Publication*, 2007.

Tong, J., Attribute based access control: a new access control approach for service oriented architectures, *Workshop on New Challenges for Access Control*, Ottawa, Canada, 2005.

Wang, J., Yu, A., Zhang, X., and Qu, L., A Dynamic Data Integration Model Based on SOA, *2009 ISECS International Colloquium on Computing, Communication, Control, and Management*, pp. 196-199. IEEE, 2009.

WU, J., and XI, C., The Study on Service Oriented Access Control Model, *Second International Conference on Information and Computing Science*, IEEE, 2009.

Yuan, E., and Tong, J., Attributed Based Access Control (ABAC) for Web Services, *IEEE International Conference on Web Services* (ICWS'05), 2005.