An Authorization Mechanism for Access Control of Resources in the Web Services Paradigm

Rajender Nath
Department of Computer Science & Applications,
Kurukshetra University, Kurukshetra,
Haryana, India

Gulshan Ahuja
Tilak Raj Chadha Institute of Management. &
Technology,
Yamunanagar,
Haryana, India

Abstract— **With the increase in web based enterprise services, there is an increasing trend among business enterprises to migrate to web services platform. Web services paradigm poses a number of new security challenges, which can only be realized by developing effective access control models. The fact that the enterprises allow access to the resources through web services requires development of access control models that can capture relevant information about a service requester at the time of access request and incorporate this information for making effective access control decisions. Researchers have addressed many issues related to authentication and authorization of web services requests for accessing resources, but the issues related to authorization work and identity based access are still poorly addressed. Authors of this paper focus on providing an extended approach to capture relevant information about a service requester and establish a certain level of trust so that amount of authorization work required for accessing any resource is reduced and service requests are served in an efficient manner. Compared with existing access control mechanisms, the proposed mechanism has reduced the amount of authorization work required for accessing resources across varied domains.**

Keywords- Web Services; Access Control; Authentication; Authorization.

I. INTRODUCTION

Web services are loosely coupled applications which use protocols like SOAP (Simple Object Access Protocol) for message exchange, WSDL (Web Services Description Language) for interface description, and UDDI (Universal Description, Discovery, and Integration) for service discovery and communication across different security domains in a distributed environment. Currently, the most of the business enterprises try to align their business processes with the IT infrastructure support by migrating towards web service oriented architecture (WSOA). The use of identity certificates issued by trusted third parties to assign roles to users has already been addressed. Looking at the available specifications in the web service security areas like WS-Security, WS-Trust, SAML, XACML, it is understandable that developers give more emphasis on securing web services and often ignore access to the resources. In web services paradigm, a web service operation must be granted access based on the permissions and by also considering the web service invocation parameters. The aforementioned models consider an action to be a limited set of operations like read, write, execute etc. and do not consider use of invocation parameters for making access control decisions. The problem with such mechanisms is that a significant amount of authorization work is required towards verifying the credentials for each service request. Every time, a service requester sends a request to the service provider for a desired service, its identity certificates issued by the third party are verified and access control policies are evaluated before granting access to any resource. This makes the task of enforcement of access control burdensome and time consuming. To address the above problems, we propose an authorization mechanism for access control of resources in web services paradigm. The proposed mechanism considers service request invocation parameters to derive new set of parameters and uses access percentile to access any web service. Our approach presents a new means to realize the authorization in an efficient manner. The efficiency of proposed mechanism is considered in terms of reduced authorization work and time to serve any request. Our approach makes use of generalized parameters, a subset of which have to be supplied by the service requester at the time of making a request to access any resource using a web service. Another subset of parameters can be set by system administrator as per the requirements of domain in which the access is to be granted. These parameters are used to check whether access to required web service can be granted or not. The proposed approach also deals with the concept of identity tracking so that the service requests belonging to a particular service requester can be logged and the subsequent requests from the same service requester can be served with less frequency of validation checks.

This paper has been organized into IX sections. Section II discusses the state of access control for resources in web services environment. Section III discusses the related work. Section IV discusses problem formulation. Section V proposes an authorization mechanism for access control of resources in web services paradigm. Section VI presents a n example case. Section VII presents a contrast of our approach vs. existing access control approaches. Section VIII presents performance evaluation results. Section IX concludes the paper and presents future scope and improvements.

II. AN OVERVIEW OF ACCESS CONTROL FOR RESOURCES

In web services environment, service providers willing to provide services publish their web services' interfaces in public registries like UDDI. Service requesters must search and locate the web services from this publicly accessible database. Once an appropriate service has been found, the service requester can send the request to service provider for the service by encapsulating the requester's information in SOAP message using HTTP protocol. The service provider draws out the invoking parameters from SOAP messages and initiates the process to execute the appropriate methods for granting access to the required resources. Service requester and service provider may not belong to the same domain. In this case, every time service requester needs to access a resource in different domain, it has to provide its credentials for authentication and authorization. As the number of available resources and requesting entities increases, the amount of authorization work also significantly increases. The existing role based access control (RBAC) models solve this problem by providing mapping between users and roles, permissions and roles. In order to guarantee more security for resource access, researchers have developed techniques like attribute certificates (AC) based on privilege management infrastructure (PMI) [1]. PMI allows including and revoking attributes and can contain information about the privileges or roles of a user.

Every time a service requester needs to access a resource, it must provide its identity and attribute certificates, which must be validated before granting access permission. The access to resources can be made more secure by establishing certain level of trust and the amount of authorization work can be reduced by using the established trust level.

III. Related Work-Access Control Models

In 1969, the basics about access control were described formally for the first time [3] and the concepts of subjects and objects were introduced. To control the access to objects, security policies were introduced. To access any object, the security level of subject has to be more than the security level of the specified object. These types of models are termed as lattice based access control (LBAC). LBAC models are not scalable and their use is restricted in specific scenarios. To overcome these limitations, Sandhu et al. [4] in 1996 proposed the notion of Role-Based Access Control (RBAC) for enterprise applications as a powerful and generalized approach for access of enterprise resources. Later, Sanhu presented RBAC [5], which defines four reference models for RBAC as follows: RBAC0 model defines a basic RBAC system, RBAC1 model augments RBAC0 with role hierarchies, RBAC2 adds constraints to RBAC0, while RBAC3 combines RBAC1 and RBAC2. ARBAC'97 (administrative RBAC'97) [6] have three components: 1) URA97, which is concerned with userrole assignment; 2) PRA97, which focuses on permission role assignment and is a dual of URA97; 3) RRA97, which deals with role-role assignment

With the advent of web services [7, 8], the access control mechanisms became more complex. The most significant work in relation to access control in the web services was proposed by Damiani et al. [9].

Freudenthal et al. [10] presented a distributed role-based access control model (dRBAC) for dynamic coalition environments. dRBAC presents a scalable, decentralized trust-management and access control mechanism for systems that span multiple administrative domains, which utilizes PKI identities to define trust domains, roles to define controlled activities, and role delegation across domains to represent permissions to these activities.

Xu et al. [11] proposed a context aware access control model for web services. Miao Liu et al. [12] proposed an attribute and role based access control model for web services. Yizho Zhao et al. [13] proposed a model a flexible role and resource based access control model (RRBAC) which can support open and distributed environments. RRBAC is suitable for multiple security domains with different applications. RRBAC adopts role directory structure instead of role hierarchy but do not consider about authorization of requests using services.

Xu Feng et al.[14] proposed a security architecture for web services and employs a SOAP proxy for processing of SOAP messages. R Bhatti et al.[15] highlights a trust based context aware access control model for web services. Christian Emig et al. [16] proposed an access control metamodel for web service oriented architecture.

IV. Problem Formulation

A significant amount of research has been carried towards access control in web services paradigm. We have based our research on RBAC because it brings much convenience to system administrators and service requesters by introducing the concepts of role in the system. However, RBAC do not fit well for multi security domains and cannot support trust management (TM) [17] and trust negotiation (TN) [18]. Many papers have proposed frameworks to integrate RBAC with web services and addressed issues like security, context based access, attribute based access, trust establishment etc. With the increase in number of service requesters and available resources, the amount of authorization work also increases.

Security architecture [14] for web services employs a SOAP proxy for processing of SOAP messages. The architecture considers only parsing of messages and realization of a role for service access. It lacks to consider about the amount of authorization work required for web service access. The work presented by R Bhatti et al.[15] highlights a trust based context aware access control model for web services but do not consider how trust level can be established and used for access control. Christian Emig et al. [16] proposed an access control metamodel for web service oriented architecture. Their approach proposes to use an authorization verification service but do not highlight that how service requesters parameters can be utilized for reduced authorization effort.

We derive a motivation here to introduce the mechanism to use service invocation parameters like requester's credentials - identity certificates, digital signatures, object identifier etc. to establish a level of trust and conserve identity of service requests for subsequent accesses. The proposed approach uses the parameters supplied during service request and derives a new set of parameters for making access decision.

V. The Proposed Work: An Authorization Mechanism for Access Control of Resources in Web Services Paradigm

Fig. 1, presents a model for implementing the proposed approach. The authorization approach defined by the authors of this paper uses a number of modules. The purpose and use of each module is defined as follows

A. Details about modules

1) Service Initiator Module (SIM)

The model comprises of Service Initiator Module (SIM), which is the main module for initiating the process for authorization and allowing access to the required web service.

2) Identity Tracker and Verifier (IDTV)

Identity Tracker and Verifier (IDTV) module is responsible for computation and assignment of access percentile and verification of the attribute certificates. Service Provider (SP) maintains a local database of Service Requesters for maintaining information about their identity.

3) Role Based Access Control (RBAC) processor

This module is composed of Policy Evaluator (PE) for evaluation of policies and Role Mapper (RM) for mapping service requester to assigned role. RBAC module checks service requests against specified policies from the policy store and invokes appropriate role for accessing a resource. There are three functions to perform the mapping:

- (service requester SRi, role Ri): to map a service requester SRi to a role Ri.
- (role Ri, resource RSi): role assignment, to map a role to a certain resource, i.e., relating the role Ri with resource RSi
- (resource RSi, permission set PSi): resource assignment, to map a resource to a certain permission set, i.e., corresponding the resource RSi , with the permission set PSi.

4) History Buffer (HB)

Is used for temporary conservation of information about past accesses to the resources.

B. Implementation Details for Authorization Mechanism

Service Provider (SP) maintains a local database of Service Requesters for maintaining information about their identity. To access any resource through a web service, the service requester sends a SOAP message, which includes various pieces of information about service requester to the service provider. On service provider's end, Service Initiator Module (SIM) processes it before passing it to the required web service for allowing access to the resource. Each web service is assigned an access percentile which specifies a value that must be possessed by any service request for accessing a resource through a web service.

The access percentile for each web service request is computed based on number of parameters supplied during service request. These parameters in the form of records are also maintained in history buffer. Table 2 represents a set of records for calculation of access percentile.

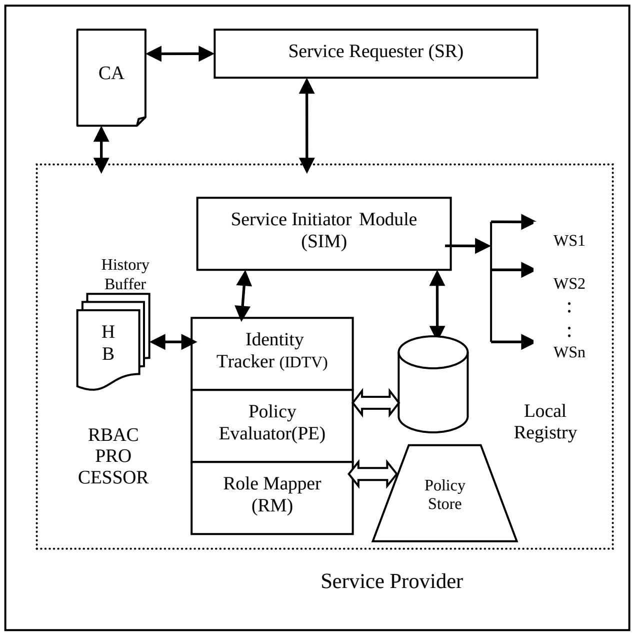

Fig. 1. Authorization architecture for access of resources using web services.

TABLE 2. RECORDS FOR ACCESS PERCENTILE

Access Parameters \ Service Requesters	IP_Addr	Id _Cert	Service _ Info	Time_ to_ Live
SR_1	192.168.0.15	AXZAB-----YPZ	WS_1, WS_3	TTL_1
SR_2	192.157.12.17	QPOST-----ORP	WS_6, WS_7, WS_9	TTL_2

Columns in the table 2 represent access parameters IP_Addr, Id_cert, Service_Info, Time_to_Live fields. IP_Addr field maintains information about IP address from where the previous requests had originated. Id_Cert field maintains information about id certificate of service requester. Service_Info field maintains information about which all services have already been accessed by a service requester. Time_to_Live field states the time period for which the record in the history buffer remains alive.

The set of parameters P_rS is chosen as a generalized set, which can be changed as per requirements. Based upon these access parameters, the access percentile is computed as described below.

Each individual parameter is assigned a weight $W_{i,}$ which is assigned by the system administrator.

IP_Addr field is assigned a weight W_{IP} according to the type of IP. For e.g static IP address may be assigned more weight than dynamic IP address.

ID_Cert field is assigned weight W_{ID} as per the trust and reputation of the CA.

Service_Info field is assigned a weight $W_{IS.}$ This field keeps track of which all services have already been accessed by a service requester in a given time frame. More is the number of times a given service WS_i has been accessed the greater is the value of weight $W_{IS.}$

Time_to_Live field is assigned a weight $W_{TTL.}$ The value of

TTL field decides that for how long the record will remain in the history buffer. The value for the weight W_{TTL} is computed as follows:

$W_{TTL} = RTE/TTL$

Where RTE is the remaining time within which the record will expire in the History Buffer (HB) and TTL is the total time to live in the HB. The value of weight will be more for frequently accessed services and will decrease if a service is accessed after some time gap.

Now, the access percentile for any service requester $SR_i,_{Access\_Percentile}$ can be calculated as

$SR_i,_{Access\_Percentile} = X_{i,,j} * W_{IP} + X_{i,,j} * W_{IP} + X_{i,,j} * W_{IS} + X_{i,,j} * W_{TTL.}$

$X_{i,,j}$ is a presence matrix where i represents any service requester SR_i value of j ranges as $1 \leq j \leq N_{AP,}$ where N_{AP} is the number of access parameters stored in HB.

$\forall SR_i \exists$ <IP_Addr, Id _Cert, Service_Info, Time_ to_ Live>

<IP_Addr, Id _Cert, Service_Info, Time_ to_ Live> $\asymp$ <0,0,0,0> iff SR_i does not exist in HB.

<IP_Addr, Id _Cert, Service_Info, Time_ to_ Live> $\asymp$ <1,1,1,1> iff SR_i exists in HB.

The SOAP message is checked for identity certificates of the service requester. The identity certificates can be obtained from a certified authority (CA) who is a trusted server in the domain, which generates the X.509 certificates. Each domain can have its own CA or if two or more domains trust each other then they can use a single shared CA. The primary responsibilities of CA include: Identification of clients requesting for issue of certificates, to Issue, archive and delete the certificates, to maintain a namespace of unique names for the certificate owners.

Reference Fig. 1., If SOAP message does not contain identity certificates the request is rejected otherwise local database of service requesters is checked to find out that requesting service requester already exist in the registry or not. If service requester exists in the registry, its access percentile is computed using stored parameters from HB, to check whether it can call the required web service for accessing a resource. Any service requester serving the request for the very first time will not be having any record in the history buffer. In this case, the service requester has to go through a complete cycle of validation process for granting access to any web resource through a web service.

Once a service requester has been authenticated and authorized, information about its service request parameters is recorded in the history buffer and in the local registry of registered users. The next time, if the same service requester needs to access a resource; its information is checked in the local registry, because the same service requester had already been authenticated and authorized, its record will be found in the local registry, after that the history buffer is accessed to compute access percentile based on available parameters.

If the service requester's access percentile is equal or more than the permitted access percentile, the request is passed on to RM for role mapping, otherwise the request is passed to PE for policy evaluation to cross verify against specified policies and thereafter the required resource can be accessed using the called web service. In case the computed access percentile is less than the permitted level of access percentile for required web service, the access is denied.

C. *The algorithms for the proposed mechanism are described in Fig. 2.1, Fig. 2.1(a) and Fig. 2.2*

The Service Provider (SP) will identify the decryption key and the elements to decrypt. Thereafter, each encrypted element is decrypted. If decryption fails a fault message is sent to the service requester. SIM carries the validity of SOAP message using the following method.

```
InitiateRequest( )
Input: Parameters from service requester
Output: allow,deny
{
If $SR_{IP}$ € Permitted$_{IP}$
{
        if $SR_i,_{Time\_Stamp}$ € Permitted$_{Time\_Stamp}$
        {
            if ( VerifySignature() )
                {
                  If ( ProcessRequest() )
                     return true;
                  else
                     return false;
                }
            else
                return false;
        }
        else return false;
}
else return false;
}
```

Fig. 2.1 Validation of SOAP message

```
VerifySignature( )
Input: - Parameters from service requester
Output:- true,false
{
Use public key ($SR_{PK}$) of Service Requester

Digital Signature → ($SR_{PK}$) → message digest

Apply hash → message digest.

Compare and match
}
```

Fig. 2.1(a) Verification of digital signature

Verification is carried to check whether the SOAP message and enclosed credentials have come from a valid service requester or not. When service provider receives a signed SOAP message, it will initiate the verification process. The public key of the service requester is used to decrypt the digital signature and retrieve the message digest. The hash algorithm is applied again to the digital contents to generate another message digest. These two message digests are compared and if they match verification is successful. If there were any changes in the digital contents the resultant message digest would differ from the original one and the verification would fail.

```
ProcessRequest( )
Input:- Parameters from HB, Records from Local Database, Parameters from service requester
Output:-true,false
{
 If $SR_{i},_{IDCert}$ exist in SOAP message
        Check → Local registry of $SR_i$ where $1 \leq i \leq n$
        If $SR_i$ found
        $SR_{i},_{Access\_Percentile}$ =ComputeAccessPercentile($SR_i$ )
                If ($SR_{i},_{Access\ Percentile}$> $WS_{i},_{Allowed\_Percentile}$ )
                        Pass SOAP message → RM
                        return true;
                end if
        elseif (ValidateCertificate($SR_{i},_{IDCert}$))
                Pass SOAP message → RBAC Processor
                Update → HB
        else return false;
else return false;
}
```

Fig. 2.2 Processing of Service Requester's Information

Once the digital signatures have been verified, the SIM proceeds to check whether identity certificates are contained within message or not.

VI. AN EXAMPLE CASE

Suppose, SecDom, be a security domain which offers online services. O_1 is the owner of a database resource system, which can provide storage service for public. O_1 registers itself for providing access to its resource through SecDom domain. After the registration, the database storage system becomes a valid resource in SecDom and the registered resource is made available through a web service. O_1 may specify many roles which are associated with different privileges. The roles include user, guest, associate_partner, admin etc. For instance, an associate partner can own the privileges of {access, read, upgrade, downgrade, delete}. Suppose a service requester wants to access a resource in SecDom domain, he authenticates himself into the domain and sends a SOAP message to the service provider, which includes service invocation parameters like requester's credentials - identity certificates and digital signatures, object identifier, security token. On service provider's end, the service request is processed for verification of supplied credentials. After that, the identity certificates are checked and validation of the certificates is carried. This is done only incase of new users or users whose request is made after a long time gap. For all those users who make request within a specific time interval, the validation of credentials can be skipped. In this way the time required to grant access to the specified resource is reduced and it also results in better bandwidth utilization.

ACL description

ACL offers a way to store the access control information. The format of ACL is chosen as follows:

acl = (role) : (resource) : (permission set)

Table 1. ACL table at service provider's end

Assigned Role	Resource Type	Permission Set
User	database storage system	{access, read, downgrade,upgrade}
Guest	database storage system	{access}
associate_partner	database storage system	{access, read, upgrade, downgrade, delete}
Admin	database storage system	{access, read, write, delete, upgrade, downgrade, create directory, delete directory, create file, delete file }

VII. COMPARISON WITH EXISTING ACCESS CONTROL APPROACHES

In this section, we present a comparison of our approach with Existing Access Control models based on Web Services. As per Table 3, we can find characteristics and features supported in our approach.

Table 3. Proposed Approach vs. Access Control Models for Web Services

Sr. No.	Feature/Characteristic	Web Services & Role Based Access Models	Our Approach
1.	Support for distributed environment	√	√
2.	Support for web service based access	√	√
3.	Resource request management	√	√
4.	Trust level assignment to web services		√
5.	Identity tracking & Management	√	√
6.	Reduced authorization work for access requests.		√

The above comparison reflects that our approach uses a mechanism to assign access percentile to web services and uses it to make access control decisions. The concept of identity tracking is used so that subsequent requests from same service requester can be served efficiently. The number of times the validation is to be carried for granting access to resources is reduced which results in reduced authorization work during serving of access requests.

VIII. PERFORMANCE EVALUATION

We consider two different scenarios for evaluation of performance of our approach.

In first scenario, we calculate time required to grant access to any resource. The measurement is carried in two ways. 1) For users who have had already requested for a resource and 2) For those users who requested for a resource for the first time. In both the cases, without our approach the time required for n service requesters would be

Following terms are defined and used for evaluation.

Let $T_{REQ} \rightarrow$ the time required to service any request made by any service requester SR_i. The time T_{REQ}, includes the round trip communication time i.e. from service requester (SR) $\rightarrow$ service provider (SP) and vice versa.

Let $T_{D,\ CERT} \rightarrow$ Time required to validate credentials of service requester from CA.

$T_{D,\ PE} \rightarrow$ Time required to evaluate associated policies.

$T_{D,\ RM} \rightarrow$ Time required to map to an appropriate role.

$T_{SRi} \rightarrow$ Total time required to serve one service request from any service requester SR_i.

Let N be the maximum number of service requests which can be made by any service requester SR_i.

$TA_{SRi} \rightarrow$ Total time required serve N service requests from same service requester SR_i

$T_{GT} \rightarrow$ Grand Total time required for granting access to all service requesters can be defined as

Let n be the maximum number of service requesters who can make a request for a resource.

We compute grand total time for first scenario as

$T_{SRi} = T_{D,\ CERT} + T_{D,\ PE} + T_{D,\ RM}$

$TA_{SRi} = N * T_{SRi}$

$T_{GT} = \sum TA_{SRi}$ where $\forall\ SR_i\ \ 1 \leq i \leq n$

In second scenario, we calculate time and effort required to grant access to any resource using our approach. The measurement is carried for users who have had already requested for a resource and for those users who requested for a resource for the first time.

Following terms are defined further and used for evaluation.

Let P be the number of times access percentile of any SR_i is sufficient to grant access and Q be the remaining attempts for which complete validation is required.

Also $P+Q \leq N$ and where $P \gg Q$ for all service requests made within allowed time period.

So the time required to service P service requests from same service requester SR_i within a specified time frame would be

$TP_{SRi} = P*T_{D,\ RM}$

And the time required to service Q service requests from same service requester SR_i would be

$TQ_{SRi} = Q* T_{SRi}$

Grand total time for second scenario would be

$T_{GT} = \sum(TP_{SRi} + TQ_{SRi})$ where $\forall\ SR_i\ \ 1 \leq i \leq n$

The above calculations reveal that once a service requester has been validated, for subsequent requests the amount of authorization work will be reduced considerably.

IX. CONCLUSION

In this paper, an approach has been proposed to handle authorization of service requests in web services paradigm. The approach makes use of the concept of identity tracking and access percentile for invoking any service. The paper has analyzed how the service request invocation parameters can be utilized for making efficient authorization verification. The performance of the proposed approach has been tested by taking a user case scenario. The results have indicated a good improvement in the performance during authorization of the request. The legacy systems that depend upon composition of services for accomplishing a particular task may require wrapping of these services through a common service interface, which may enable business process integration in an easy manner.

REFERENCES

[1] Jordi Forne, M. Francisca Hinarejos,"Web–based Authorization based on X.509 Privilege Management Infrastructure", IEEE 2003, pp 565-568.

[2] David F. Ferraiolo, Janet A. Cugini, D. Richard Kuhn, "Role Based Access Control: Features and Motivations", Proceedings of 11th Annual Conferences, December 1995, pp 241-248.

[3] B.W. Lampson,"Dynamic protection structures",AFIPS conference proceedings, 1969, pp. 27-38.

[4] Ravi S. Sandhu, Edward J. Coyne, "Role – Based Access Control Models", IEEE Computer, Feb 1996, pp. 38-47.

[5] Ravi S. Sandhu, "Role Hierarchies and Constraints for Lattice Based Access Controls", Proceedings of the 4th European Symposium on Research in Computing Security", September 1996.

[6] R. S. Sandhu, V. Bhamidiparti, Q. Munawer. The ARBAC97 model for role-based administrator of roles. ACM Transactions on Information and System Security. Vol. 2, No. 1, Feb, 1999, pp: 105-135.

[7] Web Services Architecture, http://www.w3.org/TR/ws-arch/

[8] Web Services Activity, http://www.w3.org/2002/ws/

[9] E. Damiani, S.D.C. Vimercati, S. Paraboschi and P. Samarati, "Fine Grained Access Control for SOAP e-services", Proceedings of 10th International Conference on World Wide Web , Hong Kong, 2001, pp. 504-513.

[10] E. Freudenthal, T. Pesin, L. Port, E. Keenan, V. Karamcheti. dRBAC: disttibuted Role-Based Access Control for dynamic coalition environments. In proceeding of 21th International Conference on Distributed Computing Systems. IEEE Press, 2002, pp. 128-137.

[11] Xu F., Xie J., Huang H., and Xie L., "Context Aware Role Based Access Control Model for Web Services", Proceedings of International Workshop on Information Security and Survivability for Grid", Wuhan, China, October 21-24, 2004.

[12] Miao Liu, He Qing Guo, Jin Dian Su," An Attribute and Role Based Access Control Model for Web Services", Proceedings of the Fourt International Conference on Machine Learning Cybernetics, Guangzhou, August 18-21, 2005 pp. 1302 – 1306.

[13] Yizhu Zhao, Yanhua Zhao, Hongwei Lu," A Flexible Role and Resource based Access Control Model", International Colloqium on Computing, Vommunication, Control and Management, 2008, pp. 75-79.

[14] Xu Feng, Lin Guoyuan, Huang Hao, Xie Li, "Role Based Access Control System for Web Services", Proceedings of the Fourth International Conference on Computer and Information Technology, 2004 IEEE.

[15] R. Bhatti, E. Bertino and A. Ghafoor," A Trust Based Conext Aware Access Control Model for Web Services", Proceedings of the IEEE International Conference on Web Services, California, USA, June6-9. 2004.

[16] Christian Emig, Frank Brandt, Sebastian Abeck," An Access Control Metamodel for Web Service-Oriented Architecture", International Conference on Software Engineering Advances (ICSEA 2007).

[17] X. Feng, L Jain., "Research and development of trust anagement in web security", Journal of Software 2002, Vol. 13, No. 11, pp. 2057-2064.

[18] W. H. Winsborough, K. E. Seamons, V. E. Jones," Automated trust negotiation. In DARPA Information Survivability Conference and Exposition", IEEE Press, 2000, pp. 88-102.