I.J. Information Technology and Computer Science, **2015, 12, 82-88**
Published Online November 2015 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2015.12.10

Unifying the Access Control Mechanism for the Enterprises Using XACML Policy Levels

Prof. N. Senthil Kumar
School of Information Technology & Engineering, VIT University, Vellore
Email: senthilkumar.n@vit.ac.in

Prof. Anthoniraj Amalanathan
School of Computer Science & Engineering, VIT University, Vellore
Email: aanthoniraj@vit.ac.in

Abstract—Many enterprises have intended to promote their applications with stern access control mechanism and yield the stringent authorization deployment in their individual proprietary manner. The development of this build up will result in tight coupling of authorization mechanisms within the enterprise applications. In many enterprises setup, the implicit authorization processes are embedded within the application and promote error prone accessing of requested policies. This sort of embedded authorization will let the users to carry out the specific actions without knowing the access control policy as well as its embedded setup with the help of third party involvement. But this approach has some serious effects in controlling the issues such as skipping the trust based applications, violates the policy setups and pave the way to exploit the authorized data to the end users. Many enterprises had faced serious problem in controlling its sensitive data from this implicit authorization decisions and hence decided to develop a security mechanism which can be totally controlled by centralized way of access policy. Therefore, the eXtensible Access Control Markup Language (XACML) provides a very simple and powerful remedy for authorization mechanism and for the access policy set ups.

Index Terms—EXtensible Access Control Mark up Language (XACML), security policy, Policy Enforcement Point, Policy Decision Point.

I. Introduction

Most of the modern web applications have started augmenting their security levels through high end access control systems. The proper utilization and deployment of access control over the divergent applications can provides an effective security to the selected fields and ensures the stable authorization principles on the populated resources like data, programs and services. Therefore XACML is a preferable and standardized policy language.

If we look at the security policy of the prominent organizations or the enterprises, there would be many enforcement elements to probe at various levels. Most of the enterprises gives their first priority to each enforcement policies independently and try to configure the security levels as precisely as possible. Any deviation or violation of any such enforcement will bring down the overall security policies and totally devastating the enterprise access control systems. But it is really tough to get the consolidated view of security enforcement in the entire enterprise and it often becomes too expensive and unreliable at times.

The eXtensible Access Control Markup Language(XACML) is a promising technology for developing the stable access control mechanism and providing the tight authorization decision in any enterprise applications. The XACML has derived from XML language and gives the developers an easy platform to work and enable the setups. The XACML code can be commuted to any tool or technologies and implement the policy. The generic implementation of XACML has starts with connection of both implicit and explicit authorization methods for any specific applications. This sort of setup has given the clear indication to the policy documents whether to permit or deny the client's request. The security policy for the big enterprise has hold the many access elements and serious of level enforcements.

II. Related Work

The XACML has been proposed by OASIS and a comprehensive detail of references pertaining over XACML is available at OASIS 2006. According to [Backes, et al, 2004], they had proposed algorithm for verifying the differences persist over two privacy policies and observed some implicit deviations of policy integration. In their work, they had just restricted their work only with privacy policies. Later, [Barth, et al , 2004] had extended the work and given some of the issues that are prevailing over the policy languages between two policies and came out with the observation that the two different policy levels cannot be merged dynamically and automatically as well.

In the due courses, [Zhang et al, 2005] had developed the translator that serve the translation from the conventional policy language to XACML. The Access Control (AC) policies can be verified and authenticated dynamically and guaranteed the system to yield the

Copyright © 2015 MECS

required access control policies. However, they had faced some serious of problems in integrating the policies of different levels and failed to make over from the longings. In the same year, [Anderson, 2005] has created the successful service authorization mechanism for the Web Service domain and termed it as WSPL (Web Service Policy Language). It has the inherent capability of giving the solution to specifying the policy levels, verifying the policy enabled language and apply the constraints to the policy attributes.

Then, [Fisler , 2005] had implemented the tools called Margrave which takes the input and process the XACML policies which satisfies the policy levels and policy constraints. Margrave is an automated tool that performs for policy analysis and determining the change on rules on the whole policy set up. It has specifically applied for complicated access control policies and set the matrix dynamically to cross-verify the differences.

III. XACML – Overview

XACML is a markup language and an XML based language primarily used to set the application policies, initial request and response towards the access control mechanism. The prime objective of XACML is to facilitate and offer the access rules to varied domain policies which are differed in granularities and unifying those policies into the single policy for making the access control decision in a distributed environment. The input and output of all XACML implementation follows the similar formats and on contrary, different XACML code will yield the same output on its kind. Therefore, XACML implementation gives the enterprise a stable and flexible environment wherein domain access and security level enforcement has been widely covered without any leniency and further bottlenecks. And also, XACML was originally approved by OASIS [13] and it has recommended to many commercial products like BEA Systems, Sun Microsystems, IBM etc.,

As the XACML was fully supported and recommended by OASIS, handling the policy language and access control mechanism of the application would made easier and both can well encoded in XML. In particular, the policy language determines the flow of basic access control requirements and enables the new functions, constants, data types, logics, etc to build the stringent security enforcement. Besides, the policy language takes the request or response language from the user as a query in a XML form and analyze that whether the taken action should be proceed and allowed to view the results or not. But the policy response includes the reply with the constant four actions: Permit, Deny, Indeterminate or Not Applicable. For every request, the response would be determined by the policy code to process further.

3.1 XACML Functionality

In general, there are many powerful proprietary and strong application specific markup languages looming in the market to do this sort of stuff but the values has imposed much high only in XACML prevalence. The core functionalities supported by standard XACML [5] are listed as follows:

- ***Standard:*** Since the XACML has been well organized and widely accepted by OASIS, there would be no need to register the system for every time and also no need to worry about the tricky issues pertaining in designing the new policy. Besides, XACML has offers easy deployment and well interoperate among other applications.
- ***Generic:*** A single policy set up has been widely covered to many distinguished domain applications and managing the access policy would have become easier for smooth control of security lapses. New intrusion of policies can be shunned and protected. High capability of protection would be endorsed to the applications and paves easier functioning of resource handling.
- ***Distributed:*** The XACML policy can be written and deployed in one environment (i.e., in any arbitrary locations) and other sub application can access the policy set up through it without any hassles. The consequence of this is that, instead of managing the one monolithic access policy, many users or groups will monitor the policies at appropriate intervals and the XACML has the capability to merge the results to remove any anomalies or hangovers.
- ***Powerful:*** Since there are many base languages to guard the policy extensions, XACML has the unique standard in yielding the strong and reliable policy set ups. Besides, XACML can be extended its capability to SAML and LDAP servers where the access mechanism of the resource would be quite high and protected.
- **Rich set of Standard data-types:** XACML has a very large set of built-in data-types as well as option of adding new data types. It paves ways to support not only the primitive data-types from XML, such as *String*, *Boolean*, *time*, etc. but also some distinctive data-types, such as *rfc822Name*, *x500Name*, *yearMonthDuration*, etc..

IV. The XACML Implementation

The basic operation on any application is that user likes to take some action on the available resources. To access the resource, a request has to be to sent but mostly it was well protected by the enterprise application. The Policy Enforcement Policy (PEP) is main setup which holds all the protected resources and maintains the list of protected resources with respective policy identifications [10]. Basically, the PEP can take the request on the specific parameters like requesters attributes, the question for the resources, the action to be taken over the resources and other relevant details lingering to the user request over the resources. Then the PEP will pass these request parameter to the next end which is called Policy Decision Point(PDP) which will takes these request and

analyze the PEP request parameters with specific policy decisions that whether the request which has been sought for the resource will be granted to access or not. There are two possible options that both PEP and PDP can be made available within the single application or spread over many servers. Apart from yielding the basic request or response methods, the XACML can be used for other policy decision like evaluating the policy setups and other policies pertaining against the resources.

The XACML defines four layers [2] [16] to access policy control:

- The Policy Administration Point (PAP) defines the security policies of the application and loadsthe policies in the LDAP server.
- The Policy Enforcement Point (PEP) evaluates the access control by choosing the requests and enforcing authorization decisions.
- Policy Information Point (PIP) supplies the details of the policies such as the source of attribute values, orthe data required for policy evaluation.
- The Policy Decision Point (PDP) evaluates the policy andsupply ansuitable authorization decision.

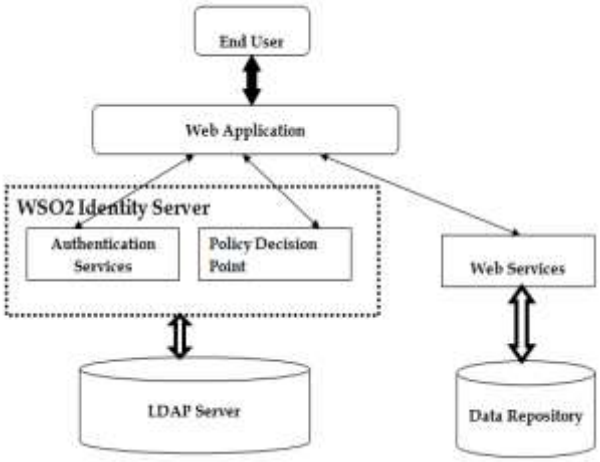

Fig. 1. The XACML Framework for Controlling the Application Level policies

4.1 The Abstract XACML Components

As mentioned earlier, the XACML syntax has the follow up of XML format. The XACML evaluation policies and its abstraction have been written purely based on XML well formed syntax. Basically, the XACML has three core policy components: PolicySet, Policy and Rule. The PolicySet is the base and root of all other XACML policies [7].

- ***PolicySet:*** A PolicySet is comprised of policy id, target elements, sequence of other PolicySets.
- ***Policy:*** A Policy is entailed with sequence of Rules, the target, and the policy combining algorithm id.
- ***Rule:*** A Rule is consists of a Target, a Condition and its access (i.e., either permit or deny).

The target in the XACML components [1][4] means that in which category, the policy has been widely applicable and indicated. In order to fetch that detail, it has the option of AnyOf conjunction and AllOf disjunction to Match the components. Each match in that classification takes only one specific category to be satisfied with the request. The typical categories of XACML parameters are subject category (e.g. users, systems, workstations), action category (e.g. read, update, delete, write), resource category (e.g. database, server, LDAP) and the environment category (SAML, J2SE, CORBA). A condition in the XACML policy is a set of propositional logic that restructures the rules. Based on the conditional value, a request can be directed into the any of the access request such as subject, action, resource and environment categories. A request can even have extra details like holding the external states (i.e, current time, temperature, etc).

4.2 XACML Evaluation Criteria

The successful evaluation of the XACML policy can be determined through the iterative process until the Match reached its true value and thereafter the based on the Match part, the component which the user has intended to access for will be granted to view. Hence, the criteria for evaluating the access policies of any application can be absolutely rely on the Condition and its Match. The Condition criterion is a set of propositional logic where each formula is determined on Boolean values either true or false or sometimes indeterminate. If the value of the Match is indeterminate, then there is an error during the flow of policy access and any more decision can't be reached during that time. An exception in XACML condition [6] is that the empty condition is treated as true always. The Rule which is underlying the condition means that, the result of any decision is applicable, not applicable or indeterminate. The policy and policyset in the XACML are evaluated by the combining algorithm which directs the access control to make correct decisions.

4.3 The Three – Valued Logic in XACML

In the XACML evaluation process [12] [17], there has been three-valued logic to determine policy decisions [8]. Therefore it defines the three-valued logic as L_3= (V_3, ≤)where the V_3 is the set (T, N, I) and N≤I≤T. For every subset S of V_3, there would be the greatest lower bound and least upper bound at S.

The evaluation criteria of XACML components to values in V3 arelisted in the following table 1.

Table 1. XACML Components evaluation criteria

V3	Match & Target Value	Condition Value	Rule & Policy
T	Match	True	Apply
N	Not Match	False	Not Apply
I	Indeterminate	Indeterminate	Cant Say

Match Evaluation

The criteria for evaluating the access policies [9] of

any chosen application can be absolutely depend on the Match value. Hence, a Match element M is an attribute value that takes the request and fulfills its condition.

The evaluation step of the Match element is as follows:

$$[M](Q) = \begin{cases} T & M \in Q \\ N & M \in Q \\ I & \text{There is an error during evaluation} \end{cases}$$

Target Evaluation

Let M be a Match and Q be a Request, the option for the components [9] [11] such as AllOf, AnyOf, Target takes the due operation in quenching the policy conditions.

- Let A be the AllOf and it can be denoted as $A = M1 \wedge M2 \wedge M3 \wedge \ldots\ldots \wedge Mn$. It can be mathematically represented as:

$$[A](Q) = [Mi](Q) \qquad Eq \qquad (1)$$

- Let E be the AnyOf and it can be denoted as $E = A1 \vee A2 \vee A3 \ldots \vee Am$. The mathematical formula for computing it is as:

$$[E](Q) = [Ai](Q) \qquad Eq \qquad (2)$$

- Let T be the Target evaluation and it have the $T = K1 \wedge K2 \wedge K3 \wedge \ldots \wedge Kn$ and the evaluation of it as:

$$[T](Q) = [Ki](Q) \qquad Eq \qquad (3)$$

Condition Evaluation

The Condition evaluation is the critical part of XACML component [13] and it gives the direction of access policy set ups. The conditional evaluation function takes the arbitrary function eval to evaluate the condition based on the request Q. The evaluation of Condition is defined as follows:

$$[C](Q) = \mathrm{eval}(C, Q) \qquad Eq \qquad (4)$$

Rule Evaluation

Let R = (*, T, C) be a Rule and Q be a Request. Then, the evaluation of Rule isdetermined as follows [9] [13]:

$$[R](Q) = \sigma([T](Q) \rightarrow [C](Q), *) \qquad Eq \qquad (5)$$

Policy Evaluation

The standard evaluation of Policy element is as follows:

Table 2. XACML policy matching

Target Value	Rule Value
Match	At least one Rule value is applicable
Match	All Rule values are not applicable
Match	At least one Rule value is indeterminate
Not Match	Don't Care
Indeterminate	Don't' Care

V. XACML PROFILES

XACML has provided various types of profiles like SAML Integration, LDAP, Digital Signature, Hierarchical Resource, Privacy, Role Based Access Control etc for exploiting access control policies in diverse applications. From those profiles, the hierarchical resource profile and resource based access control profile are widely applicable in many applications and has been extensively deployed in many organizations.

5.1 Hierarchical Resource Profile

The hierarchical resource profile is used to organize the resources in a tree based manner. Like XML, the hierarchical resource profile has the root to hold the control over all other elements or components. It can follow either tree based hierarchy where the single node takes the control or forest hierarchy where there would be multiple roots but no way interlinked with one another for form a circle. Every node in the tree or forest is treated as independent resources.

The core functionality of this profile is to control the access permission of multiple resources by enacting the single policy and getting the response of all other resources by holding the single request. The root node preserves the details of policy sets and the rules to be followed for other conditions. In this hierarchical resource policy, there would be no exploitation of rule or cross-over and enables the uniform set up throughout the application. To express the policies in the hierarchical resource, the utmost importance has to be given for discriminating which is the parent node and its appropriate child nodes. This discrimination should be made while making the hierarchy.

Besides, for representing access control policies in the hierarchical resources, the XACML supports four primary parameters:

- ***Document id:*** Identifies the resources and its associated policy.
- ***Resource Parent:*** Set the resource as parent to all other resources in that hierarchy.
- ***Resource Ancestor:*** Select the resources which are eligible for being the child resources of that structure.
- ***Resource Ancestor or Self:*** Identify any resource (mostly the requested resource) or its child resources in a hierarchical structure.

5.2 Role Based Access Control (RBAC)

The Role Based Access Control (RBAC) in XACML provides the clear path to assign the policies to a specific node or to the multiple nodes which has distinct roles. It has also offered the certain permissions to those roles and gives the specific privileges on the resources. Unlike hierarchical resource profile, we can set multiple policies to different resources and there is no single point request to get the response from all other resources.

The standard role based access control has contained the basic five elements and all of these are following XACML format:

- Users: Denotes an individual who access the resources and act as a subject in the XACML.
- Roles: Expresses as parameters to the subjects.
- Objects: The objects are same as resources in which the access controls are embedded.
- Operations: Sets the types of operations which are going to take place on the resources.
- Permissions: Gives the access rights and privileges on roles.

VI. Threats and Possible Inclusions to XACML

Although XACML has been named as the popular access control mechanism in the real world entities, it still have the deficiency of holding the privacy of the chosen applicationdomains such as authenticating the user's credentials, failed in verifying the attributes for access requests, low level category in maintaining the integrity of access policies, not supporting new techniques ensued in the given policies. Therefore, extension of XACML [6] [11] has to be improved and must able to incorporated to new intrusion of techniques. It must support some of the open source web application servers like Fedora Server8, JBoss, Application Server7, etc.

In most of the critical cases, if there were no proper authentication ensured to the application, only by some assumption, the requester can give some information which seems correct in their perspective and increases the chance of potential threat which often called as Identify Theft in the software design architecture. The identity thefts are like a malicious users issues the possible value to the access request and make the request error and track the root of the error, impersonation attack where the user act on behalf of the authorized users, etc. Therefore, without proper authentication systems, verifying the users request through authorization policies are absolute waste.

In many ways, there would have been high chance of getting the confidential data of the authorized user and later used for creating a faked profile for some attack (i.e. profile harvesting). But as of now, the security and authentication mechanism has been steeply developed and installed at various end points for keep the authorization process simple and effective. The XACML has also been a supporting factor of these measures and enables quick access to the configured systems.

6.1 Advantages of XACML

The XACML has wide range of scope and vibrant impact on many cases of policy incursion. It has given the enterprise a solidarity on policy making and solving the impeding process for long run. The following are the major advantages of using XACML:

- Through one standard access can monitor and manage many policy languages of various organizations/enterprises.
- Enterprise Administrators can save time and money due to the fact that they need not rewrite the policy level for different languages.
- Besides, the developers need not develop the new policies to the demand because they can reuse the existing standard policy languages.

6.2 Case Scenarios for Policy Integration

The policy integration has been paving way for wide range of application coordination to Web Services, Grid Systems, and Integration Services etc. To substantiate this process, we have taken two case scenarios to make it so easy to understand. They are (i) Enabling user access control policies in collaborative storage system and (ii) Enterprise Information Management.

6.2.1 Scenario 1: Storage System for Enabling User Defined Control Policies

This is just a simple Peer – to – Peer System that gives room for participating node to share the storage network and safe guard the content that has been shared between the participating peers by absolute replication of data mechanism. To make it so clear, let's take an arguments:

A data warehouse D1 want to protect and preserve its content by replicating it to many other data warehouse centers [D2, D3… Dn]. In this case, lets assume, data warehouse D2 and D3 are preparing to share the resources to the participating nodes from the storage network since they have been requesting for resources. But D2 and D3 are replicas of D1 and now the policy of these two data warehouse should be known and made global. Hence, the specification of policies according to XACML is given detailed like " Deny Override", " Permit Override", " Block Update", " Grant Privileges", etc. Hence, the global policy language define the overall policy range of the system and it further leads to absolute storage system.

6.2.2 Scenario 2: Enterprise Content Processing

Let us consider a situation where a company has decided to offshore the accounting services to different companies located in other counties. These servicing companies can fetch the much needed data from the Internet Data Service Providers and share among themselves for further manipulation and calculation of accounting. The examples of such providers are Amazon S3, iDisk, etc. In order to make the transactional process

Copyright © 2015 MECS

easier and interoperate with flexibility, we need not only set the authorization process for the chosen clients but also fix the access control policy of the company as well.

The company has to define the XACML policy range and level of each stakeholder that counter apart and propose the governing model of the whole execution. This brings the proximity between the elapsed closeness and bridge the nearness for the whole transaction that happens for the accounting process. So the enterprise must ready for setting the access control mechanism for policy language and define in the proper range of declarations.

VII. The Open Source Implementations

There are many existing implementation for enacting the right policy decisions to the enterprises and some of the implementation are open source as well as closed sources(i.e., proprietary). Among all the implementations, very few have quenching the requirements and met the standards of the large enterprises or applications.

Some of the open source XACML implementations are JBossPicketBox, SICSACML, GOSAC, XACML Light10, XACML Enterprise 11, Authorization API13 and Sun XACML [3]. Each implementation mentioned above have offered only certain features like consolidated data types to its application, specialized algorithms, well supported functions, indexing methods, attribute finder and other extended features.

If we closely looked at the organization decision point of view, the XACML access control policies have formed complex structure in many instances and each policy has been following different rules for the same. There would be always some conflicts in sharing the rules among its entire operation and lead to the major loophole for future vulnerability. To find the solutions for this hiccup, the open source implementations have satisfied the process in considerable ways and it has been treated as low level APIs [8] but very similar to standard XACML.

Hence, the Sun XACML has been chosen as the most popular choice for most of the enterprises and received large acceptance in both commercial applications and in research endeavors. Besides, the Sun XACML API supports XACML2.0 version and can parse access control policies of many applications.

VIII. Conclusion

Although the utilization of XACML to the corporate users seems new and adaptations to the standards become the strange task, it has the potential to simplify the security administrations over the enterprises. It made the task of defining the enterprise policies simple and easy to monitor. It has provide the ease of use to the security administrators to define the policy with one language and set the access control of variety of application among the enterprise. Once the XACML policy code has been furnished among the systems, then it paves the easy inclusion of authentications mechanisms and offered the great deal to include the authentication code to all its potential users. By this process, it has drastically reduce the cost for implementing the separate security concerns for its applications and affordable to interoperate with multiple applications. When the interoperability in the enterprise gives the strong control, then simultaneously it will increase the user confidence over the application and entrusted networks. Thereby, the satisfactions level of the enterprise goes high and turning the users productive at all costs.

References

[1] Aburahma, Maha, and ReinhardStumptner. “Modeling Location Attributes Using XACML-RBAC Model.” *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, 2009. 251–254.

[2] Anderson, Anne et al. “eXtensible Access Control Markup Language (XACML) Version 1.0.” *OASIS*, 2003.

[3] Ardagna, Claudio Agostino et al. “An XACML-Based Privacy-Centered Access Control System.” *Proceedings of the First ACM Workshop on Information Security Governance - WISG ’09*. ACM Press, 2009.

[4] Camenisch, Jan et al. “Credential-Based Access Control Extensions to Xacml.” *W3C Workshop on Access Control Application Scenarios, Luxembourg*. Vol. 17, 2009.

[5] Ekelhart, Andreas et al. “XML Security - A Comparative Literature Review.” *Journal of Systems and Software* 81.10, 2008: 1715–1724.

[6] Giambruno, A et al. “MagicNET: XACML Authorization Policies for Mobile Agents.” *Internet Technology and Secured Transactions, 2009. ICITST 2009. International Conference for*, 2009. 1–7.

[7] Hsieh, G. et al. “Using XACML for Embedded and Fine-Grained Access Control Policy.” *2009 International Conference on Availability, Reliability and Security* (2009).

[8] Hu, V.C. et al. “Conformance Checking of Access Control Policies Specified in XACML.” *31st Annual International Computer Software and Applications Conference (COMPSAC 2007)* 2 (2007).

[9] Hummer, Waldemar et al. “An Integrated Approach for Identity and Access Management in a SOA Context.” *Proceedings of the 16th ACM symposium on Access control models and technologies - SACMAT ’11* (2011): 21.

[10] Mazzoleni, Pietro et al. “XACML Policy Integration Algorithms.” *ACM Transactions on Information and System Security*2008 : 1–29.

[11] Mohan, Apurva, and Douglas M. Blough. “An Attribute-Based Authorization Policy Framework with Dynamic Conflict Resolution.” *Proceedings of the 9th Symposium on Identity and Trust on the Internet - IDTRUST ’10*. ACM Press, 2010. 37.

[12] Ngo, Canh et al. “Multi-Data-Types Interval Decision Diagrams for XACML Evaluation Engine.” *Privacy, Security and Trust (PST), 2013 Eleventh Annual International Conference on*, 2013. 257–266.

[13] Sun Microsystems Inc. “XACML Implementation Programmer’s Guide.” *http://sunxacml.sourceforge.net/guide.html*. 2003.

[14] Web News and Product Reviews. “XACML Access Control Markup Language.” *Wilde’s WWW Online Glossary*. 2003.

[15] X. Maroñas, E. Rodríguez, J. Delgado, “An architecture

for the interoperability between rights expression languages based on XACML", in Proc. of the 5th International ODRL Workshop, France, September 2009,
[16] C. E. Gates, "Access control requirements for Web 2.0 security and privacy", Position paper accepted to the Workshop on Web 2.0 Security and Privacy (W2SP), CA, USA, May 2007.
[17] E Fernan dez Buglioni, "Security Patterns in Practice", Designing Secure Architectures Using Software Patterns, Wiley Software Patterns Series, Wiley, 1 edition, (2013) May 28
[18] C. Ruan and V. Varadharajan, "Dynamic delegation framework for role based access control in distributed data management systems," Distributed and Parallel Databases, vol. 32,no. 2, pp. 245–269, 2014.

Author' Profiles

N. SenthilKumar: received his Master degree in Information Technology from VIT University, Vellore, India and M.Phil from Periyar University, Salem, India. .He has been working in VIT University as Assistant Professor since 2008. He has pocketed ten years of teaching experience since 2005. His research areas include Semantic Web, Web Mining, Information Retrieval, Query Expansion and Web services. He is currently holding a project on effective query expansion of semantic web and building a project over it.

Anthoniraj Amalanathan: holds both Bachelor and Master degree in Information Technology, and he is working as an Assistant Professor in VIT University, Vellore, India. Currently he is doing research in Semantic Web. Knowledge Engineering, Text Mining, Machine Learning, Open Source Programming are main interested areas to him. He is the developer and contributor of Open Source Community. He has also developed websites for some nonprofit organizations. He has delivered lot of Open Source Related talks and conducted Linux workshops.

How to cite this paper: N. Senthil Kumar, Anthoniraj Amalanathan,"Unifying the Access Control Mechanism for the Enterprises Using XACML Policy Levels", International Journal of Information Technology and Computer Science(IJITCS), vol.7, no.12, pp.82-88, 2015. DOI: 10.5815/ijitcs.2015.12.10