Managing Uncertainty in Access Control Decisions in Distributed Autonomous Collaborative Environments

Petros Belsis[1], Stefanos Gritzalis[1], Christos Skourlas[2], and Vassilis Tsoukalas[3]

[1] Department of Information and Communication Systems Engineering,
University of the Aegean, Karlovassi, Samos, Greece
[2] Department of Informatics, Technological Education Institute, Athens, Greece
[3] Department of Industrial Informatics, Technological Education Institute, Kavala, Greece
{pbelsis, sgritz}@aegean.gr, cskourlas@teiath.gr,
vtsouk@teikav.edu.gr

Abstract. Coalitions of autonomous domains gain constantly interest during the last years due to the various fields of their potential application. A lot of challenges of both academic as well as of practical nature are related with their deployment. Among else, the distributed nature of a coalition demands special focus in respect to security management. In this paper we argue about the necessity for adjustable security mechanisms towards the security management of multi-domain environments; we describe an approach that allows determination of preferences when defining access control permissions over the shared objects. We handle such preferences by encoding access control constraints using fuzzy relations and we describe a prototype security architecture that implements the basic principles of our approach.

1 Introduction

In various collaborative environments (such as ministries in e-government environments, interconnected medical domains etc), there is a need for joint access over shared resource among different organizations. The aim of these collaborative environments is to increase the capability of the participating domains to respond in various challenges without demanding excessive times. Achieving interoperability, retrieving efficiently knowledge assets in the distributed environment and managing security are among the main research challenges when building similar infrastructures; still the developed security mechanisms lack when it comes to their capability to be adaptive.

We present a method that incorporates extensions to multi-domain security models and allows determination of preferences in handling access control decisions over the shared resources for the participating domains. We describe in the framework of ongoing research work the modular components of our prototype architecture that implements policy based access control enforcement; another feature is that it allows reasoning over access requests by incorporating in the calculations fuzzy constraints. The rest of the paper is organized as follows: section 2 presents related work in context, outlining in brief differentiations from our approach; section 3 discusses some basic principles for access control enforcement in multi-domain environments.

I. Cervesato (Ed.): ASIAN 2007, LNCS 4846, pp. 261 – 267, 2007.

Section 4 presents our approach that allows determination of preferences in access control decisions for shared objects in the federated domain, using fuzzy constraints. Section 5 presents the prototype architecture that enables access control enforcement in collaborative environments. Section 6 concludes the paper, evaluating the implementation status of our approach and providing directions for further work.

2 Related Work

Barker and Stuckey [1] use constraint logic programming to express multiple access control policies; in their work they do not provide support for multiple access control restrictions, such as limitations to access objects at certain locations. In addition there is no support for determination of preferences over the access constraints.

Bonatti et al [5], propose an algebra for the creation of an access control policy out of simpler policies. In their model their language's expressiveness is analysed with respect to first order logic. They show that their language's formal semantics are equivalent to first order logic formulations. A global policy is composed out of simpler ones; in our approach instead we enable a policy bridging mechanism enabling interoperation between the constituent policies. We also define a novel technique that allows determination of preferences over specific requests or actions over the shared workspace.

In [6] a scalable solution to enable formation of coalitions over ad-hoc environments is proposed. A distributed service registry is utilized to enable interoperation between different autonomous wirelessly interconnected domains. In this approach, the management of the coalition is performed using information codified in the registry, which plays similar role as in our approach; still the proposed approach does not allow the flexibility to manage access requests that are not explicitly defined in advance.

3 Access Control Solutions for Multi-domain Environments

The Role Based Access Control (RBAC) model [2] has proved so far to be the most prominent security model; RBAC parameters can be encoded as policy expressions and can be codified in policy languages. The recorded policies are further loaded and interpreted dynamically; accordingly the policy enforcement modules reason over specific access requests. Thus, we have also adopted a policy driven approach in order to simplify and automate security management.

Considering permissions as a set of Boolean constraints associated with a given role, we can represent the security policy using constraints, each one consisting of a triplet of the following variables: the role variable, the permissions and the assets (objects) which the role is allowed to access. In a multi-domain environment where different domains share their assets, the problem of assigning privileges to roles can be cast to specifying generalised constraints (that contain tuples with RBAC variables from different domains) which have to be jointly satisfied. Typically an access control decision is defined by a tuple (Role, Object, Permission) where R is one of the available roles in the system, O is the requested object and P is an access permission (in

the UNIX© system for instance access permissions can be represented as w, r, x for write, read and execute permissions respectively). The evaluation of allowed requests - and therefore the system's access control operation - may rely on decomposition of RBAC related tasks which can furthermore be evaluated on the basis of appropriate logic expressions. Thus, in order to evaluate a state <authorised(john, write, o)|true> a number of sub-goals may be evaluated, such as: <ura(john,R_1, date)>, active(john, R_1), senior_to(R_1, R_2), pra(write,o, R_2, date)> where ura, pra, senior_to, correspond to typical RBAC expressions (user-to-role assignment, permission–to-role assignment etc)[1].

In a multi-domain environment we can consider that security tuples may be expanded so as to contain the role which originated the request, the corresponding role to the target domain, the requested object and the permission under request. Considering that all the tuples cannot be defined always in advance as new organizations join or leave, an alternative approach may rely on defining a way to express preferences over the shared objects, which define the criticality of the object and thus the willingness of a domain to share or not the resource.

4 Determining Fuzzy Relations for the Access Control Model

Access control problem formulations can be easily encoded by means of appropriate constraint representations; this is mainly due to the constraint nature of the RBAC model. Therefore allowed accesses can be represented as tuples of RBAC variables. In multi-domain environments it is not easy to describe all the possible access combinations in advance, since due to the dynamic nature of the environment new systems join and new roles and shared assets are continuously contributed to the shared environment. Security management -in contrast to the single domain paradigm- within the federated framework is much more complicated, since it is not feasible to always determine in advance all the allowed accesses. The inherent uncertainty in managing access control in these environments [11] may be treated using fuzzy relations which allow determination of the degree of satisfaction of a specific statement.

Since all the possible access combinations in multi-domain environments cannot be defined in advance and access constraints may not be evaluated on basis of Boolean expressions, fuzzy constraints may be used instead; thus, we may extend the notion of access constraints and associate them with a degree of satisfaction, expressed on a [0, 1] scale. By utilizing soft constraints it is possible to treat access control problems by encountering preferences expressed as values (k-tuples) that can be assigned to a set of variables. Therefore we can assign to each tuple a level of preference $\mu_e(u_{1,\ldots,}u_k)$ which assigns a value in a totally ordered set [0,1] [4]. Instead of an ordinary Constraint Satisfaction Problem (CSP) we can incorporate in our calculations fuzzy metrics, transforming the problem to fuzzy CSP's. More specifically, as a fuzzy CSP we can consider a list of variables $(x_{1,\ldots}x_k)$, a list of finite domains of values $(D_1,\ldots,D_k)$ and a list of fuzzy constraints $(c_{1,}\ldots\ldots,c_k)$. An instantiation $v^* \in D$ is considered as a perfect solution if all individual constraints are satisfied. An instantiation $v^* \in D$ is a best solution if the degree of joint satisfaction of all the constraints is maximal possible C $((c_1,c_{2,..},c_k)\underline{v^*})$. We consider that these preferences are encoded in a fuzzy relation R that associates each k-tuple $(u_{1,\ldots,}u_k)$ with a level of preference

$P(u_{1,\ldots,}u_k)$. $P_R(u_{1,\ldots,}u_k)$, $>P_R(u'_{1,\ldots,}u'_k)$ means that $(u_{1,\ldots,}u_k)$ is preferable $(u'_{1,\ldots,}u'_k)$. $P_R(u_{1,\ldots,}u_k)=0$ means that tuple $(u_{1,\ldots,}u_k)$ fully violates the constraint while $P_R(u_{1,\ldots,}u_k)=1$ means the constraint is fully satisfied.

We will show the applicability of our approach with an example. Considering that we want to express preferences for two available roles, R_1 and R_2 and two shared assets A and B and with 'w' and 'r' we define the two allowed permissions for these assets, we can model the problem as a fuzzy CSP (FCSP) with variables R (role), O (object) P (permission) with value-domains $\{R_1, R_2\}$, {A, B}, and {w,r} respectively.

We consider different combinations of domain variables encoded in a matrix as constraint representations (Table 1). In our case we consider two constraints that define the degree of preference for a combination of values for two (or more in a general case) problem variables. The first constraint associates roles with a preference to access the shared objects, the second associates the given objects with different types of permissions. Combinations which are totally unacceptable are not presented in Table 1 still they are encountered as not acceptable combinations during the computation of preferences.

Table 1. Encoding domain preferences by means of fuzzy constraints

Constraint	Satisfaction	R (role)	P (permission)	O (Object)
C_1	0.8	R_1		A
	0.2	R_2		A
	0.7	R_1		B
	0.1	R_2		B
C_2	0.3		W	B
	0.1		W	A
	0.4		R	B
	0.3		R	A

The legitimacy of an access request may be calculated using the preference constraints and by calculating a degree of total satisfaction for the possible combinations of values for all the problem variables. We can introduce at this point two useful metrics in order to estimate the most appropriate combinations: the appropriateness $a_i(v)$of a value $v \in D_i$ for a variable x_i is evaluated on the basis of the degree of the best possible joint satisfaction of the constraints referring to x_i and is defined as

$$a_i(v)=\max\{C((c_{i1},..,cih),\underline{v}) \mid \underline{v} \in D_{i1},\times\ldots\times D_{ik-1} \times\{v\}\times D_{ik+1} .. \times Dih\} \quad (1)$$

and the difficulty of a variable, which can be computed according to the following formula [3]:

$$d_i = \sum_{v \in D_i} a_i(v) \quad (2)$$

The difficulty metric can be used as an estimation of the most critical parameter, which should be instantiated first. While seeking for combinations that satisfy most the defined preferences, we utilize as a tool the aforementioned metrics. Therefore, we calculate first the difficulty of the variables under examination and accordingly we

instantiate the one which achieves the higher degree, which means that is the most critical and should be instantiated first. For this value according to equation (**1**) we choose the most appropriate value which maximizes the degree of satisfaction.

As an application scenario, we can consider the case where we have two domains that cooperate and want to share resources. In order the coalition to enable access to the shared resources, we need to establish a remote privilege management mechanism. We establish a role mapping approach which allows –under certain circumstances – access over shared resources [9][10]). The main idea behind this approach is that roles with many privileges and a high position in the role hierarchy are more likely to be granted access permissions over shared objects, even if these permissions have not been explicitly defined. We first calculate the difficulty for the variables R, O, P that participate in the two defined constraints by identifying the domain values and their degree of preference. For the role R variable as it can be seen from table 1 we have two possible instantiations and the maximal values for each instantiation, according to equation (1) are: 1 for R_1 and 0.2 for R_2, achieving thus a value of d=1.2. For variable O (object) we have as maximal values from Table 1, 0.8 for object A and 0.7 for B giving thus a value d=1.5 (for the difficulty). Similarly for the P variable (permission) we achieve for the possible values a difficulty of d=0.7. Therefore we instantiate first the Object variable assigning the value with higher preference: A. Accordingly, from the remaining variables according to equation we proceed by considering only the remaining combinations that contain this selected value A for the instantiated variable. The next variable to be instantiated is Role and as most appropriate value is R_1 which satisfies better the constraint. In a similar manner we conclude that the most preferred access action is read over the shared object. Therefore the most preferable allowed tuple is $<R_1, DB_1, r>$. We have in brief thus showed that by encoding preferences we can define which accesses are most preferred over which objects and for specific roles with higher privileges and a position higher than others in the role hierarchy.

5 Access Control Enforcement Architecture

We have proceeded in implementing a prototype that implements the basic operational principles of our model. The policy management architecture consists of the following modules (Fig. 1):

- The authentication module which evaluates the user credentials and provides through a single-sign-in procedure a SAML [8] assertion that allows interaction with all the modules within the coalition framework.
- The Policy Decision Point (PDP) that evaluates the requests according to the given policy.
- The Policy Enforcement Point (PEP) that actually implements access control enforcement.
- The policy mapping repository that stores all the necessary information to allow policy interoperation.
- The constraint solving module that loads the constraints and calculates the degree of satisfaction for a given constraint and facilitates the policy decision for requests that their legitimacy is not defined in advance.

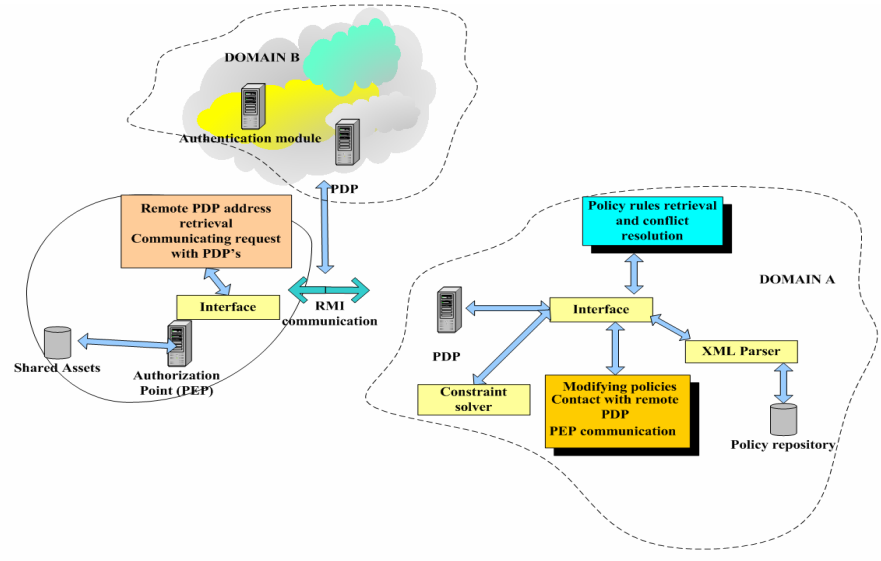

Fig. 1. Generic access control enforcement system architecture

An example usage scenario operates as follows: A user (originating in domain B) logs in the system and acquires an authentication credential which is issued as a SAML compliant assertion and which is recognizable by all the domains and allows interaction within the coalition framework; thus the operation of the framework is based on a single sign-on process which simplifies the authentication procedures for all the participating domains. In order to acquire access to shared resources, the request is formulated through the PEP interface which is implemented entirely in Java. The PEP operates using software modules that are partially provided by the XACML framework [7] and other modules that we developed for use in our multi-domain framework; accordingly, the PEP creates a XACML request (encoded in XML form). The request is sent to the PDP for further evaluation. The PDP software module is also built in Java. It primarily invokes a XACML compatible parser and isolates the access request message payload. Next it checks the request against the local policy (stored in the policy repository) to determine if the request should be authorized. In case the request comes from a role that originates in a remote domain, the PDP queries the coalition management registry and identifies whether the remote role is invoked in the coalition. This is done by sending a request to the cooperating PDP's using the Java RMI protocol. Last, the PEP receives a XACML reply message from the PDP's and enforces the decision.

6 Conclusions

Policy management in distributed collaborating environments is a challenging task, confronting with various research and technical challenges. We have presented a method that allows determination of preferences over access constraints for coalitions

of autonomous systems and have tested the validity of our approach through a prototype implementation. Specific attention has been given to the design principles of our prototype architecture so as to retain an interoperable and scalable character.

We have tested the validity of the approach by directing requests from three different domains (subnets) where each domain comprised of a three level hierarchy, with three roles for each domain. The initial performance results of the prototype were very promising. We are currently working on expanding the functionality of our prototype architecture, especially by integrating different commercial constraint solvers.

References

1. Barker, S., Stuckey, P.: Flexible Access Control Policy Specification with Constraint logic programming. ACM Trans. Inf. Syst. Secur. 6(4), 501–546 (2003)
2. Sandhu, R., Ferraiolo, D., Kuhn, R.: The NIST model for role-based access control: towards a unified standard. In: RBAC 2000. Proceedings of the Fifth ACM Workshop on Role-Based Access Control, pp. 47–63. ACM press, New York (2000)
3. Ruttkay, Z.: Fuzzy constraint satisfaction. In: Proc. 3rd IEEE International Conference on Fuzzy Systems, pp. 1263–1268 (1994)
4. Dubois, D., Fargier, H., Prade, H.: The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. In: Proc. IEEE International Conference on Fuzzy Systems, pp. 1131–1136. IEEE Computer Society, Los Alamitos (1993)
5. Bonatti, P., di Vimercati, D.C.S., Samarati, P.: An algebra for composing access control policies. ACM Trans. Inf. Syst. Secur (TISSEC) 5(1), 1–35 (2002)
6. Mukkamala, R., Atluri, V., Warner, J.: A Distributed Service Registry for Resource Sharing among Ad-hoc Dynamic Coalitions. In: Proc. of IFIP Joint Working Conference on Security Management, Integrity, and Internal Control in Information systems. LNCS, Springer, Heidelberg (2005)
7. XACML Extensible access control markup language specification 2.0, OASIS Standard (March 2004), available at http://www.oasis-open.org
8. Hughes, et al.: Technical Overview of the OASIS Security Assertion Markup Language (SAML) V1.1.OASIS, http://xml.coverpages.org/saml.html
9. Joshi, J.B.D., Bhatti, R., Bertino, E., Ghafoor, A.: Access Control Language for Multi-Domain Environments. IEEE Internet Computing 8(6), 40–50 (2004)
10. Belokolsztolszki, A., Eyers, D., Moody, K.: Policy Contexts: Controlling Information Flow in Parameterised RBAC. In: POLICY 2003. Proc. of the 4th Int. Workshop on Policies for Distributed Systems and Networks, pp. 99–110. IEEE Press, Los Alamitos
11. Hosmer, H.: Security is fuzzy!: applying the fuzzy logic paradigm to the multipolicy paradigm. In: Proceedings on the 1992-1993 Workshop on New Security Paradigms (Little Compton, Rhode Island, United States), pp. 175–184. ACM Press, New York