Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 179 (2007) 75–86
www.elsevier.com/locate/entcs

Partial and Fuzzy Constraint Satisfaction to Support Coalition Formation

Petros Belsis,[1] Stefanos Gritzalis[2] and Sokratis K. Katsikas [3]

Information and Communications Systems Engineering
University of the Aegean
Karlovasi Samos, GR 83200, Greece

Abstract

The creation of dynamic coalitions is a challenging task, seen from a security perspective. Due to the presence of conflicting requirements and specifications, the policy negotiation and policy merging processes call for the use of efficient techniques to resolve ambiguities. Constraints and constraint programming on the other hand, are useful means for representing a wide range of access control states and access control problems. In this paper we utilize constraints to represent access control policies in a multi-domain environment. In contrast to monolithic (crisp) constraint satisfaction techniques, we extend the applicability of constraints for access control, by examining soft constraints and partial constraint satisfaction. We also introduce a security framework based on fuzzy constraints that allows the determination of preferences for the participating domains.

Keywords: Coalitions, fuzzy constraints, policies, Role Based Access Control (RBAC)

1 Introduction

The emergence and rapid proliferation of networked infrastructures introduce new challenges to the integration of Information Systems. Coalitions between autonomous systems are often formed between organizations that jointly work under a common framework (ex. Ministries in e-government infrastructures, interconnected hospitals in e-healthcare environments), in order to enable access over shared resources [11]. Security under these circumstances is a major concern, since heterogeneity, different policy specifications and diverse restrictions emerge for each domain. It is also apparent that under these circumstances, conflicts are expected to emerge. The formation of such coalitions and their security management are time-consuming and error prone, if we rely more on human intervention and less on the use of flexible

[1] Email: pbelsis@aegean.gr
[2] Email: sgritz@aegean.gr
[3] Email: ska@aegean.gr

1571-0661 © 2007 Elsevier B.V. Open access under CC BY-NC-ND license.
doi:10.1016/j.entcs.2006.08.032

methods and automated tools [8].

Instead, we propose a framework that facilitates security management using constraints. Constraints are an important aspect of Role Based Access Control Models, which currently attracts considerable attention in the security research area. Various access-control restrictions and security related parameters can be formulated using constraint based representations. Particularly in a multi-domain environment, security management is harder to implement, since local policies introduce additional constraints leading to several types of conflicts; it has also been proved that the problem of interoperation among multiple policies can be considered as an instance of the satisfiability problem [3,10], which is known to be NP-complete. Therefore, in cases where multiple constraints may lead to dead-ends, partial constraint satisfaction techniques can provide alternative ways to find an acceptable solution.

The contribution of our work is on the following: we show the applicability of partial constraint satisfaction methods as a support tool for conflict resolution. We also introduce a flexible security framework, based on the combination of fuzzy constraints; through this framework domains in a multiple policy environment may define their preferences over shared resources. Therefore, the administrative overhead of the system can be minimized significantly, without violating critical constraints.

The rest of the paper is organized as follows: Section 2 discusses partial constraint satisfaction techniques and briefly discusses the use of constraints for access control. Section 3 introduces our fuzzy constraint framework and provides a detailed example on the utility of this framework to resolve policy conflicts. Section 4 discusses related work in comparison with our approach. Section 5 concludes the paper and provides directions for future work.

2 Constraints for Role Based Access Control (RBAC) Specification

A constraint satisfaction problem (CSP) consists of a set of problem variables, a set of domain values, which can potentially be assigned to these variables, and a set of constraints specifying which combinations of values are acceptable. Informally, we can consider a constraint as a combination of acceptable values for a set of problem variables. Constraints are an important aspect of RBAC. RBAC regulates the access of users to information and systems resources, on the basis of tasks that users need to execute within the system limits. A complete RBAC model includes the following variables and functions:

- The sets U (users), R (roles), P (permissions) and S (sessions)
- User to role assignment $UA \subseteq U \times R$: $U \rightarrow 2^R$
- Permission to role assignment $PA \subseteq P \times R : R \rightarrow 2^P$
- A mapping of sessions to a single user assignment US: $S \rightarrow U$
- A mapping from sessions to the set of roles associated with each session $S \rightarrow 2^R$
- A partial ordering $RH \subseteq R \times R$, represented by the symbol: $\geq$, which defines

role hierarchy. $R_1 \geq R_2$ implies that R_1 inherits permissions from R_2.

We can therefore consider $U = \langle U_1, U_2, .., U_n \rangle$ the set of users, which map to a set $R = \langle R_1, R_2, ..R_m \rangle$ of roles, and we can also consider a set $O = \langle O_1, O_2, .., O_k \rangle$ of shared resources. Additionally, access attributes may be considered members in a totally ordered set $A = \langle w, x, r, wx, .., wrx \rangle$ (combination of values w, r, x as denoted in UNIX©notation). We are interested in forming constraint specifications which are triplets of the form $< R, O, A >$.

In multi-domain environments, we are interested in assigning privileges to users belonging to another domain. This raises complexity, since classifying permissions independently for each user for his (her) and other domains may significantly increase the number of entries in an access matrix (depending on the number of participating domains and shared resources). Instead we adopt the solution of policy mappings [2], that allow the determination of corresponding roles from one domain to another. Specifically, we introduce a mapping process $\mathrm{F}(Ri, Ok, Ai) \rightarrow (R_j, O_g, A_l))$ that maps roles R_i from one domain to roles R_j from other domain. In order to ensure authorized accesses, the global policy that emerges from merging the local policies has to be compliant with restrictions originating from the participating domains. We will attempt to provide a framework that resolves such conflicts, while reducing the administrative overhead (without violating any critical constraints).

2.1 Problem formulation - Shared resources access example

Through the forthcoming paragraphs we will use an example of a (non-critical) conflict for an access control problem in multi-domain environments. For the description of the problem as well as for its solution, we will use a qualitative description of constraints. In Section 4 we will extend our framework by incorporating fuzzy constraints in the same example. Different types of conflicts as well as a more formal description of other possible types of conflicts (separation of duty, etc) are not covered due to space limitations; however, they can be treated in a similar manner. We will consider the case of two interconnected domains attempting to establish encrypted communication through IPSec [12]. According to the policy mappings predefined by coalition administrators, a remote role from domain B is assigned to domain A as (superior) Role R (Fig.1).

According to RBAC principles, R may inherit permissions from R_1 or R_2. Now considering this classification of the remote user from domain B, we want to establish a way to (semi-)automatically assign access permissions in order for him/her to be able to access the shared resources. Due to restrictions imposed by IPSec local policies (such as local firewall rules) such a task may be subject to additional constraints. For example, in IPSec a local firewall may deny the establishment of a channel between the two domains if remote access is attempted through an encrypted channel. If this happens, then the local policy restrictions do not allow interoperation between the two domains. We assume that in domain A two databases are maintained: DB_1 and DB_2, with DB_1 holding data more sensitive than DB_2. Role R_1 may be allowed to access (read) Database DB_2 and DB_1 (the second database should not be allowed to be viewed or altered remotely). In this

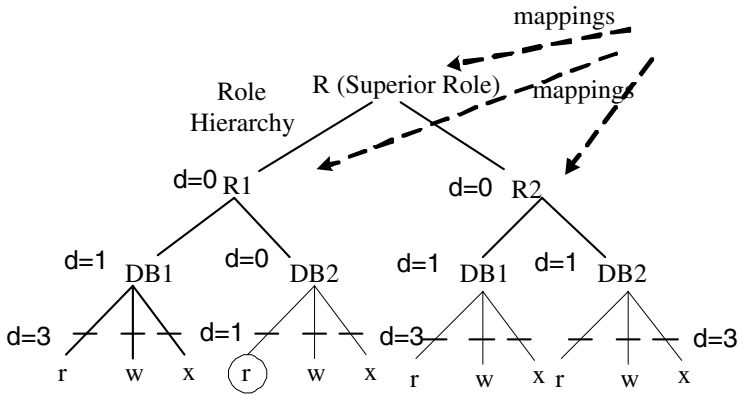

Fig. 1. A policy mapping example between two interconnected medical domains.

case access to both is denied by local firewall rules due to the restrictions imposed by IPSec. Additionally, Role R_2 may not access DB_2 remotely and should not (strict restriction) gain access to DB_1. The remote access restriction on all types of files also holds for Role B.

2.2 Partial satisfaction techniques for overcosntrained problems

In order to find acceptable combinations of the form $\langle R, O, P\rangle$, the first choice would be to use a classical algorithm such as backtracking and perform a search examining all the possible value combinations (in order to associate access rights with the shared objects for the remote role of Domain B). It is easy for someone to verify that all combinations fail, since we are referring to remote, encrypted access attempts. (the dashes over permissions in Fig.1 indicate failure of all combinations). Instead of exhaustive searching for a perfect solution that violates no constraints, we can use an alternative to backtracking approach; branch and bound technique [6] is looking for a solution that satisfies no less than a predetermined bound N (N can dynamically change during the search). In the branch and bound search algorithm, the distance parameter measured by N can be set initially according to a-priori knowledge (or according to domain's preferences) and defines the preference to satisfy no less than N constraints. During the search for a solution a search path consists of a set of assigned values over the domain variables of interest. The search path leading to the most recently chosen value for a variable is the current search path. In algorithm 1 each role in the hierarchy of remote domain B is checked against the existence of corresponding roles on Domain A; accordingly for the predefined available mappings an assignment of Objects and Permissions is performed, storing always the solution found so far that violates as fewer constraints as possible. N, S and Best-solution are global variables in the algorithm, containing the necessary and sufficient bounds (domain's preferences) and the best solution found so far, during the iterative repetitions of the algorithm.

Our approach calculates for all pairs of roles for both collaborating domains the values $< R_i, Rj, O, P >$, thus calculating for each role on a domain the possibility to access resources from other domains (based on the grounds of maximal constraint

Algorithm 1 *Step 1: For each role r_i (r_i is a role belonging to the remote domain)*
Step 2: For each role r_j (rj is a role belonging to the target domain)
Step 3: If mapping_between_roles_exists ($r_i \rightarrow r_j$) then Call Classify_PA_BB_S (role-hierarchy-path, Distance, DomainA-roles, Objects, Permissions, Values) [PA_BB_S:Classify_based_on_PArtial_Branch_and_Bound_Search_and_backtrack] [Backtrack Search: Partial Searches in order to try combinations of values over the search path and associate Access rights (permissions / privileges) with objects]
Step4: Return
Step 5: Subroutine Classify_PA_BB_S (Search_path, Distance, Variables, Values) ([Variables: the Domain variables of interest] [Values: Values assigned to the variables] [Search_path: a Set of assigned values over the domain variables] [Dimension: The number of constraints violated by the specific combination of values] [S_Bound: Dynamically computed in each iteration bound])
Step 6: If Variables=nil then [Values have been assigned to all variables in Search-path]
Step 6:Best-solution ← Search-path, N ← Distance
Step 7: If N≤ S_Bound then return 'FINISHED' [Satisfactory solution was found]
Step 8: Else return "KEEP - SEARCHING" [repeat with another value for the last variable assigned to Search-path]
Step 9: Else if Distance =N then Return "KEEP - SEARCHING" [Search-path was extended to assign values for remaining variables that do not violate more constraints]
Step 10: Else [try to extend Search-Path] Current-value ← (first value in Values) New_Distance ← Distance
Step 11: Try choices in Search-Path from first to last, as long as New_Distance<N
Step 12: If choice is inconsistent with Current-value then New_Distance ← New_Distance+1
Step 13: If New_Distance < N and Classify_PA_BB_S (Search-path plus current-value,New_Distance, Variables minus first variable, Values of second variable in Variables) = 'FINISHED' then return 'FINISHED' [Search-path sufficiently extended]
Step 14: else [check for another value] return Classify_PA_BB_S (Search-path, Distance, Variables, Values minus current value)

satisfaction, which provides a form of optimization). The advantage of branch and bound is that it does not need to search all possible pairs and that it can stop when a satisfactory solution is detected (thus achieving better response times for the overall system performance).
The algorithm crosses the search tree, by moving down to the lowest level of the tree, each level corresponding to a problem variable. A set of assigned values to the problem variables consists of a search path. The term "distance" refers to the number of constraints violated by a specific combination of values. In our example, by the time we chose to assign to role R the permissions of role R_2 (since a superior role inherits the permissions of a minor role) and by the moment we attempt to provide access to DB_1, we have a constraint violation, leading to the assignment d=1. Next, by attempting to classify permissions (one level below at the search path) over the resources, we have d=3, since DB_1 should not be accessed or modified remotely by any role, and role R_2 should not be eligible at all to access the specific resources. N is used to store the number of inconsistencies in the

best solution. As the branch and bound search proceeds, a better solution is found violating one single constraint (R_1, DB_2, r) where only the encrypted remote reading privilege is violating the domain's policy). By relaxing this (non-critical) constraint, the aforementioned RBAC policy integration approach seems to be able to get a satisfactory solution (in Fig. 1 this solution is indicated by a circle). Therefore, by applying partial constraint satisfaction techniques it is possible to achieve solutions to the multiple policies paradigm (excluding critical policy restrictions). We have to note also that this technique does not guarantee that the best solution will be found; Depending on the circumstances often immediate answers are required; for example a policy decision is subject to time restrictions. In such a case the algorithm provides the best solution found (within pre-specified time intervals). It is also not definite that a solution will be found; in worst case the search times are no better than backtracking (exponential).

3 Fuzzy constraints

In contrast to crisp constraints, soft constraints allow determination of preferences between values (k-tuples) that can be assigned to a set of variables [7]. These preferences may be considered as members of a totally ordered (fuzzy) relation, that assigns to each tuple a level of preference $\mu_e(u_1, .., u_k)$ in a totally ordered set [0,1]. As a fuzzy constraint we can consider a mapping from a domain ($D = D_1, ., D_k$) to the [0,1] interval. For a fuzzy constraint c the number $c(v_1, .., v_k)$ denotes "how well" the tuple $(v_1, .., v_k)$ satisfies the constraint.

We can extend therefore the notion of a CSP to incorporate fuzzy preferences: as a fuzzy CSP we can consider a list of variables $(x_1, .., x_k)$, a list of finite domains of values $(D_1, ., D_k)$ and a list of fuzzy constraints $(c_1, ., c_k)$. An instantiation $v^* \in$ D is considered as a perfect solution if all individual constraints are satisfied. $v^* \in$ D is a best solution if the degree of joint satisfaction of all the constraints C$((c_1, c_2, .., c_k)\underline{v^*})$ is maximal [5]. By using soft constraints we can determine multiple ways to handle preferences.

We assume that these preferences are encoded in a fuzzy relation R that associates each k-tuple $(u_1, .., u_k)$ with a level of preference P$(u_1, .., u_k)$. $P_R(u_1, .., u_k) > P_R(u_1^{'}, .., u_k^{'})$ means that $(u_1, .., u_k)$ is preferable over $(u_1^{'}, .., u_k^{'})$. $P_R(u_1, .., u_k) = 0$ means that tuple $(u_1, .., u_k)$ fully violates the constraint while $P_R(u_1, .., u_k) = 1$ means the constraint is fully satisfied.

3.1 Fuzzy relations

Fuzzy restrictions are an alternative formalism to describe fuzzy constraints, offering the ability to express prioritized constraints. They offer the possibility to model priorities -similar to preferences- expressed by levels in the scale [0,1]. A coefficient a_c expresses the priority degree of each constraint C and indicates the degree to which C must be satisfied. $a_c = 1$ means the constraint has to be fully satisfied, while $a_c = 0$ means it can be totally ignored. Therefore a fuzzy relation S on $U_1 \times .. \times U_k$ can model the pair (C, a_c) as a fuzzy relation $\mu_S(u_1, .., u_k)$=1 in case

$(u_1, .., u_k)$ satisfies the constraint C or $\mu_S(u_1, .., u_k) = 1 - a_c$ if $(u_1, .., u_k)$ violates it. In other words, μ_S is determined by whether the maximum value is achieved through satisfying the constraint or by violating it; For a soft constraint C, modeled by the fuzzy relation R, the pair (C, a_c) is represented by the fuzzy relation $\mu_S(u_1, .., u_k) = max(1 - a_c, \mu_R(u1, .., uk))$ [5].

Of great value is also the ability to treat concurrently multiple constraints. Two operations can be defined under this context: combination and projection. Given two subsets $W = \{w_1, ., w_k\}$ and $Y = \{y_1, ., y_i\}$ of the sets of variables $(x1, ..x_k)$, where $W \subseteq Y$ and a fuzzy relation T restricting the possible values of Y, then the projection of T on W is a fuzzy relation $R = T_{\downarrow W}$, defined by $\mu_R(w1, ., wk) = sup_{\{(u_{y1}, , u_(yh))/(u_{y1}, , u_{y2})\downarrow W=(u_{w1}, , u_{wk})\}} \mu_T(u_{w1}, , u_{wk})$ where $(u_{w1}, ..u_{wk})$ denotes the restriction of $(u_{y1}, ..u_{yi})$ on W. Informally, the fuzzy relation μ_R denotes to what extent a partial instantiation $(u_{w1}, .., u_{wk})$ of Y can be extended to a complete instantiation of Y that satisfies T. This is very important in case we have first instantiated the constraints of interest and we want to extend the least important constraints so as to satisfy (partially) the given problem to the highest degree.

The combination $T = R \otimes S$ of two fuzzy restrictions R and S, restricts the possible values of two sets of variables X and Y over the possible values of $W = X \cup Y$. It is defined by $\mu_T(u_{w1}, .., u_{wk}) = min\{(\mu_R(u_{w1}, .., u_{wk}) \downarrow X), (\mu_R(u_{w1}, .., u_{wk}) \downarrow Y)\}$. Typically the outcome of $\mu_{(R1 \otimes R2 \otimes R3 \otimes .. \otimes Rm)}(u_1, .., u_n)$, estimates to what extent the combination $(u_1, .., u_n)$ of values satisfies jointly the constraints. Therefore it enables us to transform preference levels on constraints into preference degrees on the possible solutions.

In addition we may consider the set of individual constraints as a decomposition of a fuzzy global relation $\rho = R1 \otimes R2 \otimes R3 \otimes .. \otimes Rn$, restricting the combination of values that may be assigned to the set of variables $(x_1, ..x_n)$. Even if there is no correlation in the set of constraints $\{R_1, R_2, .., R_m\}$, ρ implies a restriction between the acceptable values for a variable, no matter what values are assigned to other variables. In most cases there is an implied variation on values that can be assigned to other variables: $\rho^{\downarrow\{x_i, x_j\}} \subset \rho^{\downarrow\{x_i\}} \otimes \rho^{\downarrow\{x_j\}}$.

3.2 Towards fuzzy solutions

The solution of fuzzy constraint problems in most of the cases emerges as an extension of a partial solution, that instantiates the values in the given variables sequentially in such a manner that the given instantiation satisfies all the defined constraints. The notion of partial satisfaction is of primary importance within the context of fuzzy constraint problems. Selection criteria for constraint satisfaction can be the instantiation of the most critical values first, or alternatively the most constrained values first.

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. It is defined as $a_i(v) = max\{C((c_{i1}, .., c_{ih}), \underline{v}) | \underline{v} \in D_{i1} \times, \times D_{ik-1} \times \{v\} \times D_{ik+1} .. \times D_{ih}\}$. We can also measure the difficulty of a variable, according to the formula $d_i = \sum_{v \in D_i} \alpha_i(v)$[4]. This metric can be used as an estimation of the most

critical parameter, which should be instantiated first. While looking for a best solution we first instantiate variables with a limited set of appropriate values, in order to apply branch and bound techniques (which keep track of the best so far known good solution). By calculating the degree of satisfaction of an existing partial solution, we continue exploring only further solutions that achieve higher degree of satisfaction. All partial instantiations for which the degree of satisfaction does not exceed the best solution found so far are then excluded from further consideration.

3.3 Applying fuzzy constraints for access control

Con - straint	Satisfa - ction	R (role)	P (permi - ssion)	O (Object)
C_1	0.8	R_1	w	
	1	R_1	r	
	0.7	R_1	x	
	0.1	R_2	w	
	0.2	R_2	r	
C_2	0.5	R1		DB_1
	0.8	R1		DB_2
	0.1	R_2		DB_1
	0.2	R_2		DB_2
C_3	0.4		w	DB_1
	0.8		w	DB_2
	0.5		r	DB_1
	0.8		r	DB_2
	0.5		x	DB_1
	0.7		x	DB_2

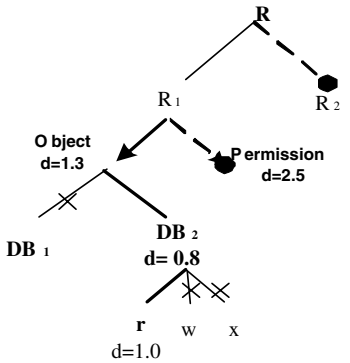

Fig. 2. a(left). Expressing preferences over constraints b(right). Calculation of values to reach the best solution.

We now re-consider the role - permission assignment problem of Section 2. We will model the problem as a FCSP with variables R (role), O (object) P (permission) with value-domains $\{R_1, R_2\}$, $\{DB_1, DB_2\}$, and {w,r,x} respectively. We have defined a matching preference according to different combinations of variables, which is represented in (Fig. 2a) (some combinations which are totally unacceptable are not represented). As already discussed, the problem is over-constrained and there is no exact solution; we can consider partial solutions only.

We will utilize as measures the appropriateness and difficulty of a variable that were described in paragraph 4.2, in order to calculate optimal solutions that satisfy the given constraints to the highest extent. Therefore we calculate for the domain variables starting from the role variable R: $a_R(R_1) = 1$, $a_R(R_2) = 0.2$, $d_R = 1.2$. For variable P (permissions), we have: $a_p(r) = 1$, $a_p(w) = 0.8$, $a_p(x) = 0.7$ $d_p = 2.5$ while for variable O (objects to be accessed): $a_o(DB_1) = 0.5$ and $a_o(DB_2)$=0.8 giving a $d_O = 1.3$. Hence, the most critical variable R that achieves lower value for the difficulty metric is first instantiated getting the value R_1, which is the value that satisfies best the constraint.

Next, among the two remaining variables, the most critical needs to be instantiated. Since there has been a selection for R, the search space for the remaining values has been reduced so as to include combinations that include the R_1 choice for the R selection (Fig 2a). Therefore, for the remaining two variables we have: $a_P(w) = 0.8$, $a_P(r) = 1$, $a_P(x) = 0.7$, with difficulty $d_P = 2.5$ and $a_O(DB_1) = 0.5$, $a_O(DB_2) = 0.8$ with difficulty $d_O(O) = 1.3$. From the last calculation it is obvious that the next variable to be instantiated is Object (O) (since the difficulty for this variable is lower) and the most appropriate value (Object) to be assigned to the already selected R_1 value (for the role variable) is DB_2.
We have achieved so far to automatically classify R_1 to be most possible to access DB_2, which satisfies better among the two choices the constraint; the next step is to check for inconsistencies with the possible combinations of permissions. We can see that the most acceptable solution is r, which achieves higher degree of satisfaction. Therefore we conclude that the most satisfactory combination is the triplet $\langle R, O, P\rangle$ $\langle R_1, DB_2, r\rangle$ (Fig. 2b). The total satisfaction degree of the achieved solution is given by the product combination principle $C_{prod}((c_1, .., c_n), \underline{v}) = \prod_{i=1}^{n} c_i(v_i)$. This metric estimates to what extent a given set of values satisfies the total set of constraints. In our case the achieved total degree of satisfaction is 0.8.

3.4 Prototype evaluation

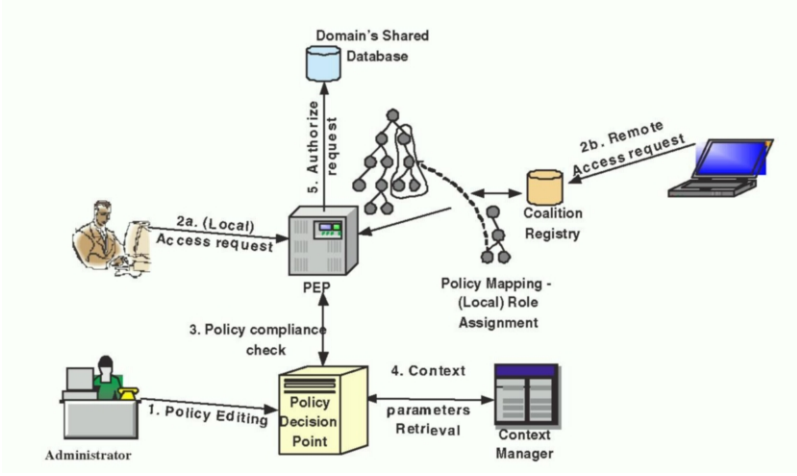

Fig. 3. Access Control architecture. The sequence of messages following a request from a remote or local domain is listed in execution order

In this section we briefly describe our prototype implementation architecture. Our basic authorization module builds upon the XACML [14] operational principles. It consists of the following entities (Fig. 3): The Policy Enforcement Point (PEP) which grants access to roles, the Policy Decision Point which reasons over a specific access request after evaluating the requestor's credentials and the request according to the available policy and the Context Manager (CM) which are responsible for collecting and sending to the PDP context related attributes, such as domain specific information.
We have implemented a special purpose registry that stores the policy mappings

and the preferences of the domains codified as numeric entries in a matrix. This registry is distributed as suggested in [11], [9] in order to avoid introducing a single point of failure. In brief, the overall operation of this multi-domain authorization framework, functions as follows: The policy administrator edits the policy and makes it available to the domain, through the Policy Decision Point (PDP). When a request for a resource appears (Fig 3), its consistency has to be validated with the local security policy prior to execution. In case of a request from a remote domain, the available mappings and the domain preferences are retrieved. A calculation of the fuzzy parameters is performed, as described in section 3.3. Next, each request (from the same or from remote domain) is directed to the Policy Enforcement Point (PEP). The request is constructed in an appropriate XML message and directed to the Policy Decision Point (PDP). Prior to the validation of the request, the Context Manager sends additional subject, resource, action and environment attributes to the PDP. Accordingly, the request is validated from the PDP and a response message is sent to the policy enforcement point (PEP), which handles the details about providing authorization to the requester.

The fuzzy decision module that calculates the criticality of constraints, presents to the administrator conflicts that achieve high satisfaction degrees (and therefore do not constitute critical conflicts). It can thus facilitate the administration of the coalition by rejecting immediately all the critical violations and by requesting further treatment for remote requests that are close to satisfying most of the locally imposed restrictions.

4 Related work

The importance of constraints for RBAC representation has been recorded recently in the relevant security literature.

Barker and Stuckey [1] apply constraint logic programming to express policies and present an easy to implement technique to represent 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 (incorporated in our approach). They also do not discuss issues of partial constraint satisfaction in the case of presence of diverse domain restrictions; moreover, they do not discuss the possibility to determine preferences over constraints.

Khurana et al. [8] define a model for the dynamic management of coalitions based on the RCL 2000 language. Coalition formation is performed as a round robin negotiation where domains make proposals about the management of shared coalition assets resources. A coalition access control matrix is formulated, that keeps records of allowed accesses; the matrix is being modified during the negotiation process and as intermediate system states are formed. Conflict resolution techniques are not discussed. Our work, focuses mainly on resolving non-critical conflicts in a secure manner with minimal human intervention.

In [13] Shafiq et al. define a policy merging algorithm that allows the determination of a global policy, based on a merging process of the individual access control

policies. For conflict resolution they define an Integer Programming (IP) based approach. In their work a global policy is formed as a sum of all roles and role hierarchies of constituting domains; this makes it hard to reflect policy updates, since the policy merging algorithm requires polynomial time. In our work, policy updates are easily integrated in the registry, while there is support to define domain preferences through fuzzy relations.

Bonatti et al [4] propose an algebra for the creation of an access control policy out of simpler policies. In their model, the expressiveness of their language is analysed with respect to first order logic. They show that the formal semantics of their language are equivalent to first order logic formulations. Our work, instead, builds a model that allows the determination of domain preferences by means of fuzzy expressions.

In [15] a flexible framework is proposed that combines subpolicies in a hierarchical manner. This framework allows the determination of safe release paths and provides support for conflict resolution by defining a number of policy operators. Our work, instead, builds upon constraints instead of logic programs, while introducing flexibility by using fuzzy constraints.

In [11], a scalable solution supporting the dynamic formation of coalitions is proposed, utilising a distributed service registry, similar to the coalition registry introduced in our approach. Our approach extends the functionalities of this approach by codifying the domain preferences in a matrix (stored at the registry) and calculating dynamically the degree of satisfaction of constraints, based on the values of this matrix.

5 Conclusions

The multi-domain policy formulation process is a complex task, subject to the presence of multiple -and of diverse characteristics- restrictions. In order to support coalition formation and to resolve conflicts, a model based on partial constraint satisfaction has been introduced. This framework has been extended using fuzzy constraints, which allow the determination of domain preferences and prioritization over constraints. We have additionally illustrated the validity and applicability of our framework by applying it to an RBAC-driven example. A prototype architecture that builds upon standardised languages and utilises principles of our framework, has also been described in this paper.

We are currently working on expanding the ability of our model to cover a wider range of constraints. We also plan to measure the performance of the resolution procedures in the presence of multiple constraints, by using a large number of access request queries from different domains as input .

References

[1] Barker, S. and P. Stuckey, *Flexible access control policy specification with constraint logic programming*, ACM Transactions on Information Systems Security (TISSEC) **6** (2001), pp. 501–546.

[2] Belsis, P., S. Gritzalis and S. Katsikas, *A scalable security architecture enabling coalition formation between autonomous domains*, in: *5th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT05)*, 2005, pp. 560–566.

[3] Bharadwaj, V. and J. Baras, *Towards automated negotiation of access control policies*, in: *4th IEEE International Workshop on Policies (IEEE Policy)*, 2003, pp. 77–86.

[4] Bonatti, P., S. D. C. diVimercati and P. Samarati, *A modular approach to composing access control policies*, in: *7th ACM Conference on Computer and Communications Security (CCS '00)*, 2000, pp. 164–173.

[5] Dubois, D., H. Fargier and H. Prade, *The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction*, in: *IEEE International Conference on Fuzzy Systems*, 1993, pp. 1131–1136.

[6] Freuder, E. and R. J. Wallace, *Partial constraint satisfaction*, Artificial Intelligence **8** (1992), pp. 21–70.

[7] Kaburlasos, V. G. and V. Petridis, *Learning and decision-making in the framework of fuzzy lattices*, in: L. Jain and J. Kacprzyk, editors, *New Learning Paradigms in Soft Computing*, Physica-Verlag GmbH, Studies in Fuzziness and Soft Computing series, Heidelberg, Germany, 2002 pp. 55–96.

[8] Khurana, H., V. Gligor and J. Linn, *Reasoning about joint administration of coalition resources*, in: *International Conference on Distributed Computing Systems*, 2002, pp. 429–439.

[9] Malatras, A., G. Pavlou, P. Belsis, S. Gritzalis, C. Skourlas and I. Chalaris, *Deploying pervasive secure knowledge management infrastructures*, International Journal of Pervasive Computing and Communications **1** (2005), pp. 265–276.

[10] McDaniel, P. and A. Prakash, *Methods and limitations of security policy reconciliation*, in: *IEEE Symposium on Security and Privacy*, 2002, pp. 73–87.

[11] Mukkamala, R., V. Atluri and J. Warner, *A distributed service registry for resource sharing among ad-hoc dynamic coalitions*, in: *IFIP 11.1 & 11.5 Joint Working Conference on Security Management*, 2005, pp. 319–336.

[12] *Rfc 2401:security architecture for the internet protocol*, http://rfc.net/rfc2401.html.

[13] Shafiq, B., J. Joshi, E. Bertino and A. Ghafoor, *Interoperation in a multidomain environment employing rbac policies*, IEEE Transactions on Knowledge and Data Engineering **17** (2005), pp. 1557–1577.

[14] *Xacml extensible access control markup language specification 2.0*, http://www.oasis-open.org.

[15] Yao, C., W. Winsborough and S. Jajodia, *A hierarchical release control framework*, in: *IFIP 11.1 & 11.5 Joint Working Conference on Security Management*, 2005, pp. 121–140.