A Logic for Multi-Domain Authorization Considering Administrators

Zeinab Iranmanesh, Morteza Amini, and Rasool Jalili
Network Security Center, Department of Computer Engineering,
Sharif University of Technology, Tehran, IRAN
{iranmanesh@ce., m_amini@ce., jalili@} sharif.edu

Abstract

In multi-domain environments, authorization policies for each administrative domain are determined by either one administrator or through cooperation of multiple administrators. Proposed logic-based models for multi-domain environments' authorization neither consider an administrator as the legislator of a policy in policies' representation nor specify the domain of a policy explicitly. Considering legislators in policy specification provides the possibility of presenting composite administration and utilizing administrators' characteristics in policy analysis such as conflict resolution. In this paper, we propose the syntax, proof theory, and semantics of a logic in which administrators are considered in authorization policies' specification, composite administration is presented, and each authorization policy is explicitly associated with some administrative domains. We also claim that the logic is sound. The presented logic is based on modal logic and utilizes two calculi named the calculus of administrators and the calculus of administrative domains. A case study of the logic usage is presented.

1. Introduction

The usage of multi-domain environments (hereafter we refer to as MDEs) is continuously increasing. MDEs constitute autonomous administrative domains. In each domain, several resources exist which may be shared among domains in large scale applications. Authorization policies regarding the resources of a domain are stated by one administrator or multiple administrators' cooperation. When a subject submits a request associated with a domain, possibly supported by one or more credentials, it must comply with the domain's authorization policies if it is to be granted [16]. Important properties of MDEs include the following:

- Authorization policies are created, stored, and managed in a dynamic and distributed fashion by cooperation among several independent administrators [16, 19].
- MDEs accommodate complex interactions, which may involve authorization policies that utilize delegations, roles, and groups [5, 21].
- Domains are typically heterogeneous; *e.g.* they have diverse restrictions [3, 19].

The properties raise specific requirements for authorization policies' representation in MDEs including the following:

- More flexible, distributed, and declarative approach to authorization representation [4, 16, 23].
- More expressive representation supporting heterogeneous or unknown policies [4, 22].
- Representation controlling interference of different authorization policies [4].
- More robust validation and analysis of policies [8].

Logic has been used to represent authorization policies in the literature due to its ability to fulfill most of the requirements. In particular, logic provides:

- Precise and non-ambiguous representation.
- Sufficient expressiveness, flexibility, and declarativeness for representation [20].
- Reasoning about authorization policies and proving properties on them [4, 23].
- Logical transformations of representations [20].

Some research [4, 16, 22] has used logic to represent authorization policies in MDEs. However, proposed models have not considered an administrator as the legislator of an authorization policy in its representation which can be used in policies' analysis and enforcement. Indeed, the research has not specified the administrative domain of an authorization policy explicitly. In this paper, we propose a logic that considers the inclusion of administrative domains and also administrators in the representation of MDEs' authorization policies. Both domains and administrators can be either primitive or composite. The definition of a prominence relation between

administrators is provided in the logic. The potential conflict among authorization policies can be resolved on the basis of the prominence relation. Our proposed logic is based on modal logic and utilizes two calculi named the calculus of administrators and the calculus of administrative domains.

The rest of this paper is organized as follows. In the next section, research related to the usage of logic in multi-domain authorization is reviewed. In section 3, a broad overview of the proposed logic is stated. The main logic, its two accommodated calculi, and its other related topics are explained in section 4. A case study is explained in section 5; and finally, conclusions are summarized in section 6.

2. Related Work

Many researchers have been attracted to use logic in presenting their minded knowledge and reason about it. Some researches have been done in using logic to represent authorization policies in MDEs. Various types of logic such as first order logic, modal logic (*e.g.* temporal logic), default logic, and stratified logic have been utilized in the representation of MDEs' authorization policies.

Some efforts have been put into specifying common abstract concepts such as roles, groups, and delegation [2, 13, 15]. Abadi, *et al.* [2] presented a calculus for access control in distributed systems. The specification of composite requesters, access control lists, role, group, and unlimited delegation have been proposed in the calculus. Howell and Kotz [13] presented a restricted version of delegation stated before in [2]. They also described the constructs of SPKI on the basis of local namespaces' logical specification. Some has focused on specifying implemented systems [1, 5, 11, 12, 17, 23]. Bowers, *et al.* [5] suggested a number of mechanisms for consumable credentials' enforcement in a distributed authorization system based on linear logic. Zhang, *et al.* [23] presented a logic-based policy specification of usage control. Using an extension of Lamport's temporal logic of actions (TLA), they developed a logical specification of UCON presented before by Park and Sandhu.

Woo and Lam [22] presented a general framework for authorization in distributed systems. A logical language for specifying *policy bases* proposed in the framework. The main drawback of the approach is that it is not decidable. Jajodia, *et al.* [14] presented a logical language for authorization specification (ASL). Every authorization specification is a stratified datalog program. Access control checking can be performed in linear time with respect to the number of rules in authorization specification. Some ideas have been presented to specify a relatively complete set of useful authorization scenarios when respecting decidability [16, 18]. Li, *et al.* [16] presented a logic-based authorization representation with delegation in distributed systems. The delegation logic focuses on explicit treatment of delegation depth and delegation to complex principals. In the logic, under some restrictions, inferencing can be computed in worst case polynomial time. Some researches have used intuitionistic logics to integrate more policy specification and its enforcement [6, 10].

Dai and Alves-Foss [7] introduced an engineering process for distributed logic-based authorization policy development. The process includes formal specification, verification, testing and integration. In order to realize policy engineering, logic programming technologies were used. Bonatti, *et al.* [4] considered composition of authorization policies that may be independently stated. They proposed an algebra of security policies together with its formal semantics. Also, it has been illustrated how to formulate complex policies and reason about them. Freudenthal, *et al.* [9] proposed a distributed role-based access control for systems that span multiple administrative domains. One of the most important features of this model is third-party delegations related to the *speaks for* relationship of Li's Delegation Logic. This feature allows privileged entities to create roles and designate other entities to give out these roles.

As can be seen, a number of models have been presented for the representation of various types of MDEs' authorization policies; however, none of these models considers the concepts of policies' legislators and domains in policies' representation. Consideration of legislators and domains (focused in this paper) provides the possibility of utilizing them in policy analysis and enforcement which has a strong potential usage in several real world applications.

3. Overview

Our proposed logic utilizes two calculi defined as *the calculus of administrators* and *the calculus of administrative domains*. The calculus of administrators is defined as a formal system whose language L_M contains proper structured administrators. Administrators may be primitive or composite. Every administrator represents a corresponding real world's authority legislating authorization policies. The calculus of administrative domains is also described as a formal system; its language L_D includes well formed administrative domains' formulae. The calculus formalizes various circumstances of domains. Our proposed logic utilizes these two calculi in order to

describe authorization statements. An authorization statement is a policy legislated by an administrator(s) and is related to some domains. The logic semantics is presented using the standard Kripke model. The soundness of the logic is proved. Also, a case study for the usage of the logic is explained.

4. The Logic for Multi-Domain Authorization

4.1. The Calculus of Administrators

We consider administrators in the specification of policies as policies' legislators by proposing a calculus for representing administrators; the calculus used later in the logic of policies. The calculus is capable of expressing possible combinations of administrators and reasoning about them. In MDEs, there are two types of administrators (as legislators): *primitive* and *composite*. Each primitive administrator is a potential single legislator and a composite administrator is a combination of primitive and/or composite administrators.

The calculus of administrators is a formal system $M = (A_m, \Omega_m, I_m)$ whose components are defined as follows:

i. A_m is a non-empty, finite, and distinct set of elements called primitive administrators and are typically shown as $m_1, m_2, \ldots$;

ii. Ω_m is a set of three functions called combinatory operators; the functions consist of: Conjunction (&), Disjunction (|), and Delegation (*);

iii. I_m is a finite set of calculus axioms explained later completely.

Depending on the precise rules of formulas' construction, the left parenthesis, "(", and the right parentheses, ")", may be necessary. The calculus functions get two primitive or composite administrators as their input and their output is a composite administrator.

The language of M, L_M, is the set of all administrators including primitive and composite administrators. Inductively, L_M is defined as the smallest set such that:

i. Every primitive administrator m_i is in L_M.

ii. If m and m' are in L_M, then so are $(m \,\&\, m')$, $(m \mid m')$, and $(m * m')$.

$m \,\&\, m'$, $m \mid m'$, and $m * m'$ are composite administrators. $m \,\&\, m'$ is used when m and m' legislate jointly, $m \mid m'$ is used when either m or m' legislates a policy, and $m * m'$ is used if m legislates as an agent of m'.

The administrators' calculus axioms determining the characteristics of the calculus functions are as follows:

(A1) L_M is closed under &, |, and *.

(A2) &, |, and * are idempotent in a wide sense. (A function is idempotent in a wide sense if it allows one of its identical operands to be deleted; and, an operand to be duplicated using it as the operator).

(A3) & and | are commutative.

(A4) &, |, and * are associative.

Considering the above characteristics, administrators' related algebraic structures can be distinguished, including:

- Administrators form a *Semilattice* under & and |.
- Administrators form a *Band* under *.

The axioms related to the distributivity property of the proposed calculus functions are as follows:

(A5) $m \,\&\, (m' \mid m'') \equiv (m \,\&\, m') \mid (m \,\&\, m'')$

(A6) $m * (m' \,\&\, m'') \equiv (m * m') \,\&\, (m * m'')$

(A7) $m * (m' \mid m'') \equiv (m * m') \mid (m * m'')$

Stipulated axioms are proved to be sound according to the presented semantics.

4.2. The Calculus of Administrative Domains

We consider administrative domains in policy specification in order to identify each policy's managed domain explicitly. The calculus of administrative domains expresses an abstraction of administrative domains, their various situations against each other, and other useful properties (e.g., algebraic properties). The calculus is then used in the logic of authorization statements. Consequently, administrative domains are accommodated in policy presentation. Hereafter, a domain is called *primitive* if it is an identified domain in MDE; and, a domain is named *composite* when it is a proper composition of other domains.

The calculus of administrative domains is defined as a formal system $D = (A_d, \Omega_d, I_d)$. The system consists of the following sets:

i. A_d is a non-empty, finite and distinct set of primitive domains presented as $d_1, d_2, \ldots$;

ii. Ω_d is a set of functions applied on domains, including: top ($\top$), bottom ($\bot$), intersection ($\cap$), union ($\cup$), and complement (-);

iii. I_d is the set of calculus axioms stated later.

The left parenthesis, "(", and the right parentheses, ")", may be necessary in formulas' synthesis. $\cup$, $\cap$, and - take two domains as their input and their output being a composite domain is the inputs union, intersection, and complement respectively. $\top$ and $\bot$ are functions without input; $\top$ represents the union of all primitive domains and $\bot$ presents no domain.

The language of D is called L_D consisting of all proper structured domains and is defined inductively as follows:

i. Every primitive domain, d_i, is in L_D.
ii. $\top$ and $\bot$ are in L_D.
iii. If d and d' are in L_D, then so are $(d \cap d')$, $(d \cup d')$, and $(d - d')$.

The calculus of administrative domains' axioms related to its functions' properties are as follows:

(A8) L_D is closed under $\cap$, $\cup$, and -.
(A9) $\cap$ and $\cup$ are idempotent in a wide sense.
(A10) $\cap$ and $\cup$ are commutative.
(A11) $\cap$ and $\cup$ are associative.
(A12) $\cap$ and $\cup$ are unital due to the satisfaction of the equations $\top \cap d = d \cap \top = d$ and $\bot \cup d = d \cup \bot = d$.

According to the stated properties, a number of algebraic structures formed from L_D and the functions defined in the calculus consist of:

- L_D under $\cap$ and $\cup$ forms a *Monoid* and also a *Semilattice*.
- L_D under – forms a *Magma*.

Some of the calculus axioms are related to the distributivity property of the calculus functions over each other including:

(A13) $d \cap (d' \cup d'') \equiv (d \cap d') \cup (d \cap d'')$
(A14) $d \cap (d' - d'') \equiv (d \cap d') - (d \cap d'')$
(A15) $d \cup (d' \cap d'') \equiv (d \cup d') \cap (d \cup d'')$

Soundness of the specified axioms has been proved.

4.3. The Logic of Authorization Statements

In this paper, authorization policies are expressed using modal logic, as authorization statements. The primary contribution is the inclusion of an administrator legislating an authorization statement and an administrative domain associated with the statement in its specification. Composite administrators and various compositions of domains' situations can be stated in the logic due to the inclusion of the calculi.

The alphabet of the logic is as follows:

i. A non-empty, finite and distinct set of authorization propositions shown in the form of $p_1, p_2, \ldots$.
ii. L_M : The set of administrators.
iii. L_D : The set of administrative domains.
iv. The connectives of the logic: $\wedge$, $\vee$, $\neg$, $\rightarrow$, *leg* (legislation), ~, and $\mapsto$.
v. The left parenthesis, "(", and the right parentheses, ")".

The defined calculi have been included in the logic by accommodating L_M and L_D. $\wedge$, $\vee$, $\neg$, and $\rightarrow$ are primitive logical connectives. The modal logic connective is *leg*. The left operand of ~ is from L_M and its right operand is from L_D. Both operands of $\mapsto$ are from L_M.

The set of all proper authorization statements, *S*, is the smallest set such that:

i. Every authorization proposition, p_i, is in *S*.
ii. If m and m' are in L_M, then $m \mapsto m'$ is in *S*.
iii. If s and s' are in *S*, then so are $(s \wedge s')$, $(s \vee s')$, $(s \rightarrow s')$, and $\neg s$.
iv. If s is in *S*, m is in L_M, and d is in L_D, then $m \sim d \ leg \ s$ is in *S*.

$m \mapsto m'$ expression implies that m' dominates m; consequently, m' would own every authorization that m had. The authorization statement $m \sim d \ leg \ s$ expresses m (administrator) legislates the authorization statement s related to d (administrative domain).

4.4. Proof Theory

The logic of authorization statements' inference rules consist of:

(R1) $$\frac{s \ ; \ s \rightarrow s'}{s'}$$

(R2) $$\frac{s}{m \sim d \ \ leg \ \ s, \text{ for every } m, d}$$

In the propositional logic, R1 is known as modus ponens; in the modal logic, R2 is called necessitation.

Some axioms are proved to be valid in the logic of authorization statements, including:

(A16) If s is a tautology in the propositional logic, then s is valid in the logic of authorization statements.
(A17) $(m \sim d \ leg \ s \rightarrow s') \rightarrow ((m \sim d \ leg \ s) \rightarrow (m \sim d \ leg \ s'))$
(A18) $(m \sim d \ \ leg \ \ s) \rightarrow \neg(m \sim d \ \ leg \ \ \neg s)$
(A19) $m \,\&\, m' \sim d \ \ leg \ \ s \equiv (m \sim d \ \ leg \ \ s) \wedge (m' \sim d \ \ leg \ \ s)$
(A20) $m * m' \sim d \ \ leg \ \ s \equiv m \sim d \ \ leg \ \ (m' \sim d \ \ leg \ \ s)$

(A21) $((m \sim d \ \ leg \ \ s) \vee (m' \sim d \ \ leg \ \ s)) \rightarrow (m \mid m' \sim d \ \ leg \ \ s)$

(A22) $m \sim d \cup d' \ \ leg \ \ s \equiv (m \sim d \ \ leg \ \ s) \wedge (m \sim d' \ \ leg \ \ s)$

(A23) $m \sim d - d' \ \ leg \ \ s \equiv (m \sim d \ \ leg \ \ s) \vee \neg (m \sim d' \ \ leg \ \ s)$

(A24) $((m \sim d \ leg \ \ s) \vee (m \sim d' leg \ \ s)) \rightarrow (m \sim d \cap d' \ \ leg \ \ s)$

$\neg$ and $\rightarrow$ are formed a complete set on which basis $\wedge$ and $\vee$ can be defined; so, the following equations are established according to A17 and A18:

(Eq1) $(m \sim d \ \ leg \ \ s \wedge s') \equiv ((m \sim d \ \ leg \ \ s) \wedge (m \sim d \ \ leg \ \ s'))$

(Eq2) $((m \sim d \ leg \ s) \vee (m \sim d \ leg \ s')) \rightarrow (m \sim d \ leg \ s \vee s')$

The axioms are proved to be sound according to the proposed semantics.

4.5. Semantics

We express the semantics of the authorization statements' logic using the standard Kripke model. The Kripke-style structure for the proposed logic is presented as $M = \langle W, I, J \rangle$. The components of M consist of:

- W is the set of possible worlds.
- $I : P \rightarrow 2^W$ is an interpretation function mapping every authorization proposition to a subset of W in which the proposition is true.
- $J : M \times D \rightarrow 2^{W \times W}$ is an interpretation function mapping each pair formed from an administrator and an administrative domain to a binary relation from W to W. The administrator and administrative domain are primitive.

To determine the semantics of an authorization statement comprising an administrator and an administrative domain, the semantics of administrators and domains should be defined; J is used for this purpose. J, based on its inputs being an administrator and a domain, specifies paired possible worlds being reachable from each other through the inputs. Each possible world is considered as an authorized state in which several requests are allowed; a request is disallowed if it is explicitly forbidden or just not permitted. If an administrator m being in w knows w' reachable according to his knowledge about a domain d, then $(w, w') \in J(m, d)$ is established. An administrator's knowledge about a domain d is considered as allowable requests regarding d from the administrator's view point. The function R extends J to accept composite administrators and/or composite domains as input:

- $R(m, d) = J(m, d)$

  For a primitive administrator and a primitive domain, R and J results are the same.
- $R(m \,\&\, m', d) = R(m, d) \cup R(m', d)$

  Suppose w is a typical possible world where administrators m and m' list reachable worlds; w' is a reachable world from w through m and a domain d; and, w'' is a reachable world from w through m' and d. The union of administrators' knowledge is obtained by their conjunction. Accordingly, both w' and w'' are reachable from w through $m \,\&\, m'$ and d.
- $R(m * m', d) = R(m, d) o\ R(m', d)$

  The delegation of administrators bridges between reachable worlds according to their knowledge about some domains.
- $R(m \mid m', d) = R(m, d) \cap R(m', d)$

  By administrators' disjunction, either their common knowledge or the knowledge of one is considered in specifying reachable worlds. The smallest set is resulted from their common knowledge.
- $R(m, d \cup d') = R(m, d) \cup R(m, d')$

  The knowledge of an administrator about the union of two domains is the union of his knowledge about each of them.
- $R(m, \top) = \bigcup_{\forall d_i} R(m, d_i)$

  Where d_i is a typical primitive administrative domain.
- $R(m, d \cap d') = R(m, d) \cap R(m, d')$

  Based on an administrator's knowledge about two domains d and d', his knowledge about their intersection is the intersection of his knowledge about each of them. As both domains' allowable requests are supposed to be allowable in their common area, contradictory ones are eliminated.
- $R(m, d - d') = R(m, d) - R(m, d')$

  The knowledge of an administrator about $d - d'$ is through removing his knowledge about d' (analogously his knowledge about two domains' intersection) from his knowledge about d.
- $R(m, \perp) = R(m, d_i) - R(m, d_i)$

  Where d_i can be any primitive administrative domain. As $\perp$ is the symbol of no domain, it can be defined as the difference of any domain and itself.

The function K extends I by mapping each authorization statement to a subset of possible worlds where it is true. It is defined as follows:

- $K(p_i) = I(p_i)$

  K and I give an identical set of possible worlds if their input is an authorization proposition.
- $K(\neg s) = W - K(s)$
- $K(s \wedge s') = K(s) \cap K(s')$

- $K(s \vee s') = K(s) \cup K(s')$
- $K(s \rightarrow s') = \{w \mid if \;\; w \in K(s) \;\; then \;\; w \in K(s')\}$
- $K(m \sim d \;\; leg \;\; s) = \{w \mid \forall w'.(w,w') \in R(m,d) \;\; then \;\; w' \in K(s)\}$
- $K(m \mapsto m') = \{w \mid (if \;\; \forall w'.(w,w') \in R(m,\top) \; then (w,w') \in R(m',\top))\}$

4.6. Soundness

The authorization statements' logic is claimed to be sound. A logic is sound if:

i. Each of its axioms is valid according to the logic semantics.
ii. Its inference rules preserve the validity.

Then by induction on the length of proof, one can verify that every well-formed expression would also be valid semantically.

The axioms and inference rules of the logic are arranged in three categories: the calculus of administrators, the calculus of administrative domains, and the logic related axioms and inference rules. Each group is proved to be sound. We present soundness proof of one axiom in each group due to high volume of proofs if we want to explain all axioms' soundness proofs. In the following proofs, Suppose a typical model $M = \langle W, I, J \rangle$ and typical possible worlds $w, w' \in W$.

Soundness of A5, A6, and A7: *A5, A6, and A7 are sound.*

Proof: $m \,\&\, (m' \mid m'') \equiv (m \,\&\, m') \mid (m \,\&\, m'')$ is proved to be sound. Suppose we have $(w,w') \in R(m \,\&\, (m' \mid m''), d_i)$ (d_i can be any typical administrative domain).

$(w,w') \in R(m \,\&\, (m' \mid m''), d_i)$
iff $(w,w') \in R(m,d_i) \cup R(m' \mid m'', d_i)$
iff $(w,w') \in R(m,d_i) \cup (R(m',d_i) \cap R(m'',d_i))$
iff $(w,w') \in (R(m,d_i) \cup R(m',d_i)) \cap (R(m,d_i) \cup R(m'',d_i))$
iff $(w,w') \in R((m \,\&\, m') \mid (m \,\&\, m''), d_i)$

The distributivity property of $\cup$ over $\cap$ (as set theory operators) is used in the above proof. $m \,\&\, (m' \mid m'') \equiv (m \,\&\, m') \mid (m \,\&\, m'')$ is valid due to the generality of *M*, *w*, and w'. There are analogous justifications for the soundness proof of A6 and A7.

Soundness of A12: *A12 is sound.*

Proof: $d \cap \top = d$ is proved to be sound. Suppose we have $(w,w') \in R(m, d \cap \top)$ (*m* can be any administrator).

$(w,w') \in R(m, d \cap \top)$
iff $(w,w') \in R(m,d) \cap R(m,\top)$
iff $(w,w') \in R(m,d) \cap \bigcup_{\forall d_i} R(m,d_i)$
iff $(w,w') \in \bigcup_{\forall d_i} (R(m,d) \cap R(m,d_i))$
iff $(w,w') \in R(m,d)$

$R(m,d) \cap R(m,d_i)$ is the subset of $R(m,d)$; since for every d_i the subsets' union set is computed, $R(m,d)$ is obtained finally. The soundness of $d \cup \perp = d$ is also proved similarly.

Soundness of A22: *A22 is sound.*

Proof: Suppose we have $\models_w^M m \sim d \cup d' \;\; leg \;\; s$.

$K(m \sim d \cup d' \;\; leg \;\; s)$
iff $\{w \mid \forall w'.((w,w') \in R(m, d \cup d') \; then \; w' \in K(s)\}$
iff $\{w \mid \forall w'.((w,w') \in (R(m,d) \cup R(m,d')) \; then \; w' \in (K(s)\}$
iff $\{w \mid \forall w'.(w,w') \in R(m,d) \; then \; w' \in K(s)\} \cap$
iff $\{w \mid \forall w'.(w,w') \in R(m,d') \; then \; w' \in K(s)\}$
iff $K(m \sim d \;\; leg \;\; s) \cap K(m \sim d' \;\; leg \;\; s)$
iff $K((m \sim d \;\; leg \;\; s) \wedge (m \sim d' \;\; leg \;\; s))$

Thus, $w \in K(m \sim d \cup d' \;\; leg \;\; s)$ if and only if $w \in K((m \sim d \;\; leg \;\; s) \wedge (m \sim d' \;\; leg \;\; s))$; accordingly, $\models_w^M m \sim d \cup d' \;\; leg \;\; s \equiv (m \sim d \;\; leg \;\; s) \wedge (m \sim d' \;\; leg \;\; s)$. The axiom is valid due to the generality of *M*, *w*, and w'.

As R1 and R2 are got from the propositional logic and the modal logic respectively, they preserve validity in our proposed logic as well.

5. Case Study

In order to show the applicability of the proposed logic in real world applications, we present a case study using the logic. The case study is related to electronic meeting systems.

Typically, an electronic meeting system is based upon a network (either wired or wireless) of microcomputers situated in an electronic meeting room. Using the system, group participants perform collaborative group work. Each group participant is considered as an administrator of the system; he/she legislates policies, including authorization policies, regarding resources (mainly micricomputers and their documents) being under his/her management. There may be several resources under the administration of a single group member. Also, several participants may use a common resource and set authorization policies cooperatively. The concept of electronic meeting system has been considered in many projects such as WeBex Meeting Center and PlaceWare Conference Center. For security management of an electronic meeting system, we suggest applying our proposed logic. For this purpose, we define the following scenario.

In an instant electronic meeting system, there are four participants named *m1*, *m2*, *m3* and *m4*. The resources of the electronic meeting room include four PCs and three PDAs connected to each other using wired and wireless networks. The PCs are called *c1*, *c2*, *c3* and *c4* and the PDAs are called *a1*, *a2* and *a3*. *ci* is under the management of *mi*. *a1* is under the cooperative administration of *m1* and *m2*, *a2* is under the cooperative administration of *m2* and *m3*, and *a3* is under the cooperative administration of *m3* and *m4*. In order to determine the administrative domain of an administrator (a meeting member) explicitely, we define the administrative domain for each administrator as *di* including all rsources being under his management; *e.g. d2* is the administrative domian of *m2* and contains *c2*, *a1*, and *a2*. If two domains have some resources in common (such as *d1* and *d2*), it is assumed that the domains are under the management of both domains' administrators.

We assume that there is a central security service controling accesses throughout the electronic meeting system according to the policies at hand legislated by different administrators.

In the scenario, the set of policies available for the central security service (in the form of our proposed logic) is as follows:

$m1 \sim d1 \ \ leg \ \ (p_1 \wedge p_3)$

$m2 \sim d2 \ \ leg \ \ (p_4 \rightarrow p_5)$

$(m3 \sim d3 \ \ leg \ \ p_3) \vee (m2 \sim d3 \ \ leg \ \ p_2)$

$(m2 \,\&\, m3) \sim (d2 \cap d3) \ \ leg \ \ p_2$

$m4 \sim d4 \ \ leg \ \ (\neg p_1 \vee p_5)$

$(m2 * m1) \sim d2 \ \ leg \ \ p_7$

$(m1 \,\&\, m2) \sim (d1 \cup d2) \ \ leg \ \ (p_1 \wedge p_6)$

It is assumed that policies existing in a list are connected to each other with $\wedge$; also, the list of policies is supposed to be consistent. Each p_i is an authorization proposition implies a set of permissions. Users presenting the set $Attribute_i$ of credentials can perform the set $Action_i$ of possible actions on the set res_i of resources.

In addition to being an administrator and the legislator of a policy, meeting members are the users of each other's resources. A user presents his request concerning a resource and does not mention its domain. The central service is responsible for specifying the domain of a resource. When a user offers his request, the central security service is responsible for authorization usually assisted by an inference engine. The service inspects all existing and inferred policies; if the request is complied with a policy, it is granted; otherwise, it is rejected. If a resource concerned in a request would not be in some domains common area ($d \cap d'$), every policy regarding the resource's domain, *d*, and its two combinations with any typical domain, $d \cup d_i$ and $d - d_i$, is considered in authorization. Otherwise, policies concerning d and its combinations except $d - d_i$ are considered. Indeed, among considered policies containing a type of domains' combinations, those are selected whose legislator is a combination of the domains' administrators. Then, compliance of a request with selected policies is checked.

For instance, consider the following two requests. User *u1* presents a request whose resources are related to $d_2 \cap d_3$ and actions are permitted according to $p_2 \vee p_3$ based on its offered credentials. The request is granted due to the following inference:

$$(m3 \sim d3 \ \ leg \ \ p_3) \vee (m2 \sim d3 \ \ leg \ \ p_2) \overset{(A17),(A18),(A21)}{\Rightarrow} (m2 \mid m3) \sim d3 \ \ leg \ \ (p_2 \vee p_3)$$

User *u2* requests some actions on resources existing in d_4; permission granting for the actions' performing requires a policy containing p_7. The request is rejected because no policy is found containing p_7 and one of the domain combinations including d_4, $d_4 \cup d_i$, or $d_4 - d_i$.

6. Conclusions

In multi-domain environments, the authorization policies of an administrative domain are legislated by one administrator or multiple administrators' cooperation. In addition, policies may be associated with a predefined domain or domains' various combinations such as their intersection.

The proposed logic in this paper considers administrators as the legislators of policies in policies' representation. This approach makes the possibility of utilizing administrators' characteristics in policies' analysis *e.g.* in conflict resolution. Administrator being in a policy representation can be primitive or composite. Three styles of administrators' composition are presented. The other contribution of this paper is the explicitly and exactly defined inclusion of administrative domains in policies' representation and their association with authorization policies. Indeed, three styles of administrative domains' combination are considered. The exactly defined semantics and proof theory of the logic provides the possibility of authorization policies' representation and reasoning about them regarding their legislators and related

domains. The soundness of the logic is proved and its completeness proof is postponed as a future work.

7. References

[1] M. Abadi, "On SDSI's linked local name spaces" *Journal of Computer Security*, 1998, 6(1–2):3–21.

[2] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin, "A Calculus for Access Control in Distributed Systems", *ACM Transactions on Programming Languages and Systems*, 1993, 15(4):706–734.

[3] P. Belsis, S. Gritzalis, and S.K. Katsikas, "Partial and Fuzzy Constraint Satisfaction to Support Coalition Formation", *Electronic Notes in Theoretical Computer Science*, Elsevier, 2007, pp. 75-86.

[4] P. Bonatti, S.D.C.D. Vimercati, and P. Samarati, "An Algebra for Composing Access Control Policies", *ACM Transactions on Information and System Security*, ACM, USA, 2002, 5(1): 1-35.

[5] K.D. Bowers, L. Bauer, D. Garg, F. Pfenning, and M.K. Reiter, "Consumable Credentials in Logic-Based Access-Control Systems", In *Proceedings of the 2007 Network and Distributed Systems Security Symposium*, 2007, pp. 143-157.

[6] J. G. Cederquist, R. J. Corin, M. A. C. Dekker, S. Etalle, J. I. den Hartog, and G. Lenzini, "The audit logic: Policy compliance in distributed systems" *Technical Report TR-CTIT- 06-33, Centre for Telematics and Information Technology, University of Twente*, 2006.

[7] J. Dai and J. Alves-Foss, "Logic Based Authorization Policy Engineering", In *6th World Multiconference on Systemics, Cybernetics, and Informatics*, 2002, pp. 230-238.

[8] A. El-Atawy, "Survey on the Use of Formal Languages/Models for the Specification, Verification, and Enforcement of Network Access-lists", 2006.

[9] E. Freudenthal, T. Pesin, L. Port, E. Keenan, and V. Karamcheti, "dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments", In *22nd International Conference on Distributed Computing Systems*, 2002, pp. 411-420.

[10] D. Garg and F. Pfenning, "Non-interference in constructive authorization logic" In *Proceedings of the 19th IEEE Computer Security Foundations Workshop*, 2006, pp. 283–296.

[11] J. Y. Halpern and R. van der Meyden, "A logic for SDSI's linked local name spaces", In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, 1999, pp. 111–122.

[12] J. Y. Halpern and R. van der Meyden, "A logical reconstruction of SPKI", In *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, 2001, pp. 59 – 70.

[13] J. Howell and D. Kotz, "A formal semantics for SPKI", In *Proceedings of the 6th European Symposium on Research in Computer Security*, 2000, pp. 140–158.

[14] S. Jajodia, P. Samarati, and V. S. Subrahmanian, "A logical language for expressing authorizations" In *IEEE Symposium on Security and Privacy*, USA, 1997, pp. 31–42.

[15] B. Lampson, M. Abadi, M. Burrows, and E. Wobber, "Authentication in distributed systems: Theory and practice", *ACM Transactions on Computer Systems*, 1992, 10(4):265–310.

[16] N. Li, B.N. Grosof, and J. Feigenbaum, "A Logic-based Knowledge Representation for Authorization with Delegation", In *Proceedings of the 12th IEEE workshop on Computer Security Foundations*, IEEE Computer Society, USA, 1999, page 162.

[17] N. Li and J. C. Mitchell, "Understanding SPKI/SDSI using first-order logic", In *Proceedings of the 16th IEEE Computer Security Foundations Workshop*, 2003, pp. 89–103.

[18] N. Li, J. C. Mitchell, and W. H. Winsboroug, "Design of a role-based trust management framework", In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, 2002, pp. 114–130.

[19] J.D. Moffett and M.S. Sloman, "Policy Conflict Analysis in Distributed System Management", *Journal of Organizational Computing*, 1994, pp. 1-22.

[20] R. Ortalo, "Using Deontic Logic for Security Policy Specification", report, *Toulouse (FR) : LAAS*, 1996.

[21] M. Sloman, "Policy Driven Management for Distributed Systems", *Journal of Network and Systems Management*, Plenum Press, 1994, 2(4): 333-360.

[22] T.Y.C. Woo and S.S. Lam, "Authorization in Distributed Systems: A New Approach", *Journal of Computer Security*, IOS Press, 1993, pp. 107-136.

[23] X. Zhang, F. Parisi-Presicce, R. Sandhu, and J. Park, "Formal Model and Policy Specification of Usage Control", *ACM Transactions on Information and System Security*, ACM, USA, 2005, 8(4): 351 – 387.