A Logic for Inclusion of Administrative Domains and Administrators in Multi-domain Authorization

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. Authorization policies for an administrative domain or a composition of multiple domains in multi-domain environments are determined by either one administrator or multiple administrators' cooperation. Several logic-based models for multi-domain environments' authorization have been proposed; however, they have not considered administrators and administrative domains in policies' representation. In this paper, we propose the syntax, proof theory, and semantics of a logic for multi-domain authorization policies including administrators and administrative domains. Considering administrators in policies provides the possibility of presenting composite administration having applicability in many collaborative applications. Indeed, administrators and administrative domains stated in policies can be used in authorization. The presented logic is based on modal logic and utilizes two calculi named the calculus of administrative domains and the calculus of administrators. It is also proved that the logic is sound. A case study is presented signifying the logic application in practical projects.

1 Introduction

In multi-domain environments (hereafter we refer to them as MDEs), there are multiple administrative domains. When a subject submits a request concerning some actions on some resources, possibly supported by one or more credentials, it must comply with authorization policies of the domain containing the resource if it is to be granted [1]. MDEs' characteristics such as being dynamic, distributed, heterogeneous, and open raising the requirement of a more powerful authorization for them. Therefore, for authorization policies' representation in MDEs, a more flexible, distributed, expressive, and declarative approach is needed. Logic has been used to represent authorization policies in the literature due to its related strengths; *e.g.* logic provides reasoning facility, sufficient precision, expressiveness, flexibility, and declarativeness in representation [2], [3], [4].

Some researches have used logic to represent authorization policies in MDEs, including [1], [3], [5], [6]. However, proposed models have not considered an administrator as the legislator of an authorization policy and its administrative domain in policies' representation explicitly. In this paper, we propose a logic considering inclusion of administrative domains and also administrators in MDEs' authorization policies; an administrator and an administrative domain can be primitive or composite.

H. Jahankhani, K. Revett, and D. Palmer-Brown (Eds.): ICGeS 2008, CCIS 12, pp. 190–199, 2008.

The rest of this paper is organized as follows: in the next section, some researches related to multi-domain environments' security are 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 real world application of the logic is studied in section 5. Finally, conclusions are summarized in section 6.

2 Related Work

Multiple domains approach to security management is introduced in some papers to split the environment into several administrative domains to make distributed security management possible. The concept is used in [7] as a security framework in pervasive computing environment. Pearlman, *et al.* in [8] introduced virtual organizations (VO) and virtual communities in which collaborative activities are made through multiple institutions resource sharing. They address policy specification for shared resources in cooperative manner and policy enforcement in VOs as a key problem in these environments. The multi-domain approach is used in [9] and [10] for mobile computing environments in controlling users' access to services in different domains. Joshi *et al.* in [11] proposed XML Role-Based Access Control (X-RBAC) specification language for multi-domain environments. In XRBAC, domains cooperation and inter-domain accesses becomes possible by specifying mediation policies. A domain-based role-based access control model (RBAC-DM) has been presented by Demchenko, *et al.*, in [12] for distributed collaborative applications; however, it does not consider the cooperative approach in security management.

Some researches have been done in using logic to represent authorization policies in MDEs. Some efforts have been put into specifying common abstract concepts such as roles, groups, and delegation including [5], [13], and [14]. Abadi, *et al.* in [5] presented a calculus for access control in distributed systems. The specification of composite requesters, access control lists, role, group, and unrestricted delegation have been proposed in the calculus. Some researches have been performed to specify implemented systems including [4], [15], [16], [17], [18], and [19]. Bowers, *et al.* in [16] suggested a number of mechanisms for consumable credentials' enforcement in a distributed authorization system based on linear logic. Woo and Lam in [20] presented a general and logical framework for authorization in distributed systems. The main drawback of the approach is that it is not even semi-decidable. Jajodia, *et al.* presented a logical language for authorization specification (ASL) in [6]. Access control checking can be performed in linear time w.r.t. 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 including [1] and [21]. Some researches have used intuitionistic logics to integrate more policy specification and its enforcement including [22] and [23]. Bonatti, *et al.* in [3] considered composition of authorization policies that may be independently stated. Freudenthal, *et al.* in [24] proposed a distributed role-based access control for systems that span multiple administrative domains.

3 Overview

Two calculi defined as *the calculus of administrative domains* and *the calculus of administrators* are utilized in our proposed logic representing authorization statements. The calculus of administrative domains formalizes domains and their various circumstances. In the calculus of administrators, every administrator represents a corresponding real world's authority legislating authorization policies. An authorization statement is a policy legislated by an administrator and is related to a domain; the administrator and the domain may be either primitive or composite. The logic semantics is presented using the standard Kripke model. Soundness of the logic is proved and a case study using it is presented.

4 The Logic for Multi-domain Authorization

4.1 The Calculus of Administrative Domains

A domain is called *primitive* if it is an identified domain in MDEs; 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 ($d_1, d_2, \ldots$);
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 which will be stated later.

The left parenthesis, "(", and the right parentheses, ")", may be necessary in formulas' synthesis. $\cup$, $\cap$, and - get two domains as their input and their output being a composite domain, is the inputs' union, intersection, and complement respectively. $\top$ and $\bot$ get no input; $\top$ represents the union of all primitive domains and $\bot$ presents no domain. The language of D is called L_D constituting from well formed administrative domains; it 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 axioms regarding the calculus functions' properties are as follows:

(A1) L_D is closed under $\cap$, $\cup$, and -.
(A2) $\cap$ and $\cup$ are idempotent in a wide sense.
(A3) $\cap$ and $\cup$ are commutative.
(A4) $\cap$ and $\cup$ are associative.
(A5) $\cap$ and $\cup$ are unital due to the satisfaction of the equations $\top \cap d \equiv d \cap \top \equiv d$ and $\bot \cup d \equiv d \cup \bot \equiv d$.

The following axioms are related to the distributivity property of the calculus functions over each other:

(A6) $d \cap (d' \cup d'') \equiv (d \cap d') \cup (d \cap d'')$

(A7) $d \cap (d' - d'') \equiv (d \cap d') - (d \cap d'')$

(A8) $d \cup (d' \cap d'') \equiv (d \cup d') \cap (d \cup d'')$

Soundness of the specified axioms is proved.

4.2 The Calculus of Administrators

In MDEs, two types of administrators (as legislators) can be found out: *primitive* and *composite*; a primitive administrator is a potential single legislator; and, a composite administrator is a proper combination of primitive and/or composite administrators.

The calculus is a formal system, $M = (A_m, \Omega_m, I_m)$; its components are 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 rules of formulas' construction, "(" and ")" 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, containing properly structured administrators is defined inductively 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 $(m * m')$.

$m \,\&\, m'$ 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 axioms determining the calculus functions' characteristics are as follows:

(A9) L_M is closed under &, |, and *.

(A10) &, |, and * are idempotent in a wide sense.

(A11) & and | are commutative.

(A12) &, |, and * are associative.

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

(A13) $m \,\&\, (m' \mid m'') \equiv (m \,\&\, m') \mid (m \,\&\, m'')$

(A14) $m * (m' \,\&\, m'') \equiv (m * m') \,\&\, (m * m'')$

(A15) $m * (m' \mid m'') \equiv (m * m') \mid (m * m'')$

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

4.3 The Logic of Authorization Statements

In the logic, an administrator legislating an authorization statement and an administrative domain associated with the statement are included in its representation, composite administrators and various compositions of domains' situations are stated 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: ~, *leg* (legislation), $\neg$, and $\rightarrow$. ($\wedge$ and $\vee$ can defined based on $\neg$ and $\rightarrow$).
v. The left parenthesis, "(", and the right parentheses, ")".

The calculi are included in the logic by accommodating L_M and L_D . The modal logic connective is *leg*. Left operand of ~ is from L_M and its right operand is from L_D . 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 *s* and s' are in *S*, then so are ($s \rightarrow s'$) and $\neg s$ (and accordingly, ($s \wedge s'$) and ($s \vee s'$)).
iii. 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*.

The statement $m \sim d \ \ leg \ \ s$ implies an administrator *m* legislates an authorization statement *s* related to *d* (an administrative domain). If no administrative domain is specified for an authorization statement, the statement is valid in all defined domains.

4.4 Proof Theory

The inference rules of the authorization statements' logic consist of:

(R1) $\dfrac{s \ ; \ s \rightarrow s'}{s'}$ (The modus ponens rule)

(R2) $\dfrac{s}{m \sim d \ \ leg \ \ s, \text{ for every } m, d}$ (The necessitation rule)

The axioms proved to be valid in the authorization statements' logic are as follows:

(A16) if *s* is a tautology in the propositional logic, then *s* is valid in the logic too.
(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) $((m \sim d \ leg \ \ s) \vee (m \sim d' leg \ \ s)) \rightarrow (m \sim d \cap d' \ leg \ \ s)$

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

4.5 Semantics

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.

If an administrator *m* being in *w* knows w' reachable according to his known allowable requests regarding a domain *d*, then $(w, w') \in J(m, d)$ is established. The function *R* extends *J*, accepting composite administrators and domains as input:

$$R(m, d) = J(m, d) \tag{1}$$

For a primitive administrator and a primitive domain, *R* and *J* results are the same.

$$R(m \,\&\, m', d) = R(m, d) \cup R(m', d) \tag{2}$$

The union of administrators' knowledge is obtained by their conjunction.

$$R(m * m', d) = R(m, d) o \, R(m', d) \tag{3}$$

Delegation of administrators bridges between their known reachable worlds.

$$R(m \mid m', d) = R(m, d) \cap R(m', d) \tag{4}$$

By administrators' disjunction, their common knowledge is considered.

$$R(m, d \cup d') = R(m, d) \cup R(m, d') \tag{5}$$

The knowledge of an administrator about the union of two domains is the union of his knowledge about each of them.

$$R(m, \top) = \bigcup_{\forall d_i} R(m, d_i) \tag{6}$$

d_i is a typical primitive administrative domain.

$$R(m, d \cap d') = R(m, d) \cap R(m, d') \tag{7}$$

An administrator's knowledge about two domains' intersection is the intersection of his knowledge about each of them.

$$R(m,d-d') = R(m,d) - R(m,d') \tag{8}$$

The knowledge of an administrator about $d-d'$ is got by removing his knowledge about d' from his knowledge about *d*.

$$R(m,\perp) = R(m,d_i) - R(m,d_i) \tag{9}$$

d_i can be any primitive administrative domain.

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:

$$K(p_i) = I(p_i) \tag{10}$$

K and *I* give identical results if their input is an authorization proposition.

$$K(\neg s) = W - K(s) \tag{11}$$

$$K(s \wedge s') = K(s) \cap K(s') \tag{12}$$

$$K(s \vee s') = K(s) \cup K(s') \tag{13}$$

$$K(s \rightarrow s') = \{w \mid if \;\; w \in K(s) \;\; then \;\; w \in K(s')\} \tag{14}$$

$$K(m \sim d \;\; leg \;\; s) = \{w \mid \text{for all } w'.(w,w') \in R(m,d) \;\; then \;\; w' \in K(s)\} \tag{15}$$

4.6 Soundness

The logic of authorization statements is proved 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 proof's length, one can verify that every well-formed expression would also be valid semantically. We avoid to present soundness proof of the logic due to high volume of proofs if we want to explain them.

5 Case Study

In order to point out the applicability of the proposed logic in real world applications, we present a case study using the logic and related to grid computing environments.

Grid resources are geographically distributed across multiple administrative domains and owned by different organizations. For solving large-scale computational and data intensive problems, resources are shared among different domains; thus, creating virtual organizations (VOs). Each domain has its own security requirements

including authorization ones legislated by domain's administrators. By constructing virtual organizations, authorization policies are legislated by administrators' cooperation for their administered domains' various situations. The specified foundations of grid environments are considered in all related projects such as Globus and NASA IPG. We consider the specified concepts in a typical grid project and represent them using our proposed logic. Consider the following scenario. In a virtual organization, there are four organizations (domains) whose situation is shown in Fig. 1.

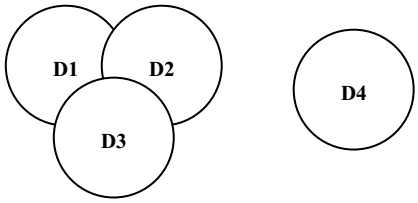

Fig. 1. Domains' situation instance

m1, *m2*, and *m3* are administrators legislating authorization policies for domains *d1*, *d2*, and *d3* respectively; and, for the domains' various combinations collaboratively. Also, *m4* administrates *d4* and specifies its authorization policies. Suppose authorization policies' list being at hand is as follows:

[AP1] $m1 \sim d1 \ \ leg \ \ p_1$ [AP2] $m2 \sim d2 \ \ leg \ \ p_2$

[AP3] $(m3 \sim d3 \ \ leg \ \ p_3) \vee (m1 \sim d1 \ \ leg \ \ p_3)$

[AP4] $m3 \sim (d3 - d2) \ \ leg \ \ p_5$ [AP5] $m4 \sim d4 \ \ leg \ \ p_2$

[AP6] $(m3 * (m1 \,\&\, m2)) \sim (d1 \cap d2 \cap d3) \ \ leg \ \ p_4$

[AP7] $(m1 \,\&\, m2) \sim (d1 \cap d2) \ \ leg \ \ p_6$ [AP8] $m4 \sim d4 \ \ leg \ \ (p_3 \rightarrow p_6)$

Each p_i is an authorization proposition implies a set of permissions. In grid environments, multi-organization is transparent to a user; thus, he doesn't state a specific domain in his request. One of services in core middleware layer of grid architecture is security service. In the case that virtual organization's authorization policies are expressed using our proposed logic, when a user offers his request, the security service is responsible for authorization. The service inspects all 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), $d \cup d_i$, 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. For instance, consider the following request. User *u1* presents a request whose resources are

related to $d_1 \cap d_3$; and, actions are permitted according to p_3 based on offered credentials. The request is granted due to the following inference:

$$(m3 \sim d3 \ \ leg \ \ p_3) \vee (m1 \sim d1 \ \ leg \ \ p_3) \overset{(A21),(A24)}{\Rightarrow} (m1 \mid m3) \sim (d1 \cap d3) \ \ leg \ \ p_3$$

6 Conclusions

In multi-domain environments, 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 provides the possibility of utilizing administrators' characteristics in authorization. Three styles of administrators' composition are presented. The other contribution of this paper is the explicitly and exactly defined inclusion of associated administrative domains in policies' representation. Three styles of administrative domains' combination are considered. Both administrators and domains can be primitive or composite. The exactly defined semantics and proof theory of the logic provides the possibility of authorization policies' representation as well as reasoning about them regarding their legislators and related domains. Soundness of the logic is proved and its completeness proof is postponed as a future work.

References

1. Li, N., Grosof, B.N., Feigenbaum, J.: A Logic-based Knowledge Representation for Authorization with Delegation. In: Proceedings of the 12th IEEE workshop on Computer Security Foundations, p. 162. IEEE Computer Society, USA (1999)
2. Ortalo, R.: Using Deontic Logic for Security Policy Specification. Report, Toulouse (FR): LAAS (1996)
3. Bonatti, P., Vimercati, S.D.C.D., Samarati, P.: An Algebra for Composing Access Control Policies. ACM Transactions on Information and System Security, 1–35 (2002)
4. Zhang, X., Parisi-Presicce, F., Sandhu, R., Park, J.: Formal Model and Policy Specification of Usage Control. ACM Transactions on Information and System Security, 351–387 (2005)
5. Abadi, M., Burrows, M., Lampson, B., Plotkin, G.: A Calculus for Access Control in Distributed Systems. ACM Transactions on Programming Languages and Systems, 706–734 (1993)
6. Jajodia, S., Samarati, P., Subrahmanian, V.S.: A logical language for expressing authorizations. In: IEEE Symposium on Security and Privacy, USA, pp. 31–42 (1997)
7. Kagal, L., Finin, T., Joshi, A.: Trust-based security in pervasive computing environments. IEEE Computer, 154–157 (2001)
8. Pearlman, L., Welch, V., Foster, I., Kesselman, C., Tuecke, S.: A community authorization service for group collaboration. In: The 3rd IEEE International Workshop on Policies for Distributed Systems and Networks (Policy 2002), pp. 50–59. IEEE Computer Society Press, Monterey (2002)

9. Au, R., Looi, M., Ashley, P.: Cross-domain one-shot authorization using smart cards. In: The 7th ACM Conference on Computer and Communications Security (CCS 2000), pp. 220–227. ACM Press, Athens (2000)
10. Au, R., Looi, M., Ashley, P., Tang Seet, L.: Secure authorization agent for cross-domain access control in a mobile computing environment. In: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 343–359. Springer, Heidelberg (2002)
11. Joshi, J.B.D., Bhatti, R., Bertino, E., Ghafoor, A.: Access-control language for multidomain environments. IEEE Internet Computing, 40–50 (2004)
12. Demchenko, Y., de Laat, C., Gommans, L., van Buuren, R.: Domain based access control model for distributed collaborative applications. In: The Second IEEE International Conference on e-Science and Grid Computing, IEEE Computer Society Press, Amsterdam (2006)
13. Howell, J., Kotz, D.: A formal semantics for SPKI. In: The 6th European Symposium on Research in Computer Security, pp. 140–158 (2000)
14. Lampson, B., Abadi, M., Burrows, M., Wobber, E.: Authentication in distributed systems: Theory and practice. ACM Transactions on Computer Systems, 265–310 (1992)
15. Abadi, M.: On SDSI's linked local name spaces. Journal of Computer Security, 3–21 (1998)
16. Bowers, K.D., Bauer, L., Garg, D., Pfenning, F., Reiter, M.K.: Consumable Credentials in Logic-Based Access-Control Systems. In: The 2007 Network and Distributed Systems Security Symposium, pp. 143–157 (2007)
17. Halpern, J.Y., van der Meyden, R.: A logic for SDSI's linked local name spaces. In: The 12th IEEE Computer Security Foundations Workshop, pp. 111–122 (1999)
18. Halpern, J.Y., van der Meyden, R.: A logical reconstruction of SPKI. In: The 14th IEEE Computer Security Foundations Workshop, pp. 59–70 (2001)
19. Li, N., Mitchell, J.C.: Understanding SPKI/SDSI using first-order logic. In: The 16th IEEE Computer Security Foundations Workshop, pp. 89–103 (2003)
20. Woo, T.Y.C., Lam, S.S.: Authorization in Distributed Systems: A New Approach. Journal of Computer Security, 107–136 (1993)
21. Li, N., Mitchell, J.C., Winsboroug, W.H.: Design of a role-based trust management framework. In: The 2002 IEEE Symposium on Security and Privacy, pp. 114–130 (2002)
22. Cederquist, J.G., Corin, R.J., Dekker, M.A.C., Etalle, S., den Hartog, J.I., Lenzini, G.: The audit logic: Policy compliance in distributed systems. Technical Report TR-CTIT- 06-33, Centre for Telematics and Information Technology, University of Twente (2006)
23. Garg, D., Pfenning, F.: Non-interference in constructive authorization logic. In: The 19th IEEE Computer Security Foundations Workshop, pp. 283–296 (2006)
24. Freudenthal, E., Pesin, T., Port, L., Keenan, E., Karamcheti, V.: dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments. In: 22nd International Conference on Distributed Computing Systems, pp. 411–420 (2002)