On the Negotiation of Access Control Policies

Virgil D. Gligor, Himanshu Khurana, Radostina K. Koleva,
Vijay G. Bharadwaj, and John S. Baras

Department of Electrical and Computer Engineering,
University of Maryland, College Park, Maryland 20742
{gligor, hkhurana, radost, vgb, baras}@eng.umd.edu

Abstract. Although the notion of negotiation has been used extensively in secure communication protocols to establish common keying states, protocol modes and services, this notion is only now emerging in the area of access control policies. In this paper, we review the motivation for access policy negotiation and we provide some examples of use in practice. In particular, we illustrate the meaning, the types, and the process of negotiation for establishing common access states and common interpretations of policy models for dynamic coalitions and large-scale access control in the Internet.

1 Introduction

Collaborative computing requires a variety of resource-access agreements ranging from those for peer-to-peer sharing of application objects and services to those for joint administration of access policies among autonomous domains of the Internet. An important characteristic of such agreements is the negotiation of common access to a set of resources reflecting the sharing preferences of the parties involved. Such negotiations typically seek agreement on a set of access properties, which represents the *common interpretation of a policy model*, and then on a *common state*, which reflects the shared accesses to resources of the parties involved and satisfies the negotiated access properties. We say that such an agreement is the result of the negotiation of access control policies.

A common interpretation of the policy model adopted by collaborating parties is essential for reaching resource access agreements, particularly when joint administration of shared resources is being sought. The common interpretation of the policy model consists of the specific properties of access authorization, management, and attribute-definition components of the policy model implemented by the collaborating parties. For example, the access management area may include the property that selective and transitive distribution of privileges requires selective and transitive revocation of privileges; or that the distribution and revocation of privileges be owner-based. Such properties must be supported by all collaborating parties for joint administration of shared resources and hence must be negotiated.

In this paper we illustrate the salient aspects of the negotiation of access control policies in the context of dynamic coalitions formed by autonomous domains

B. Christianson et al. (Eds.): Security Protocols, LNCS 2467, pp. 188–201, 2002.

of the Internet. We view dynamic coalitions as peer-to-peer networks that include resources which are jointly owned and administered by the member domains [9]. Such coalitions have two characteristics that help motivate the key aspects of policy negotiations, namely membership that (1) comprises domains managed under diverse access policies reflecting different sharing interests and (2) varies dynamically thereby ruling out off-line, one-time access policy agreements.

Previous work in the area of access control for dynamic coalitions illustrated some of the important aspects of access policy negotiations in specific settings. For example, Shands *et al.* [12] and Herzberg *et al.* [6] addressed the problem of unambiguously specifying a common access state, communicating this common state to all member domains, and committing this common access state. However, this work assumes that common access states were agreed upon by extra-technological (e.g., off-line) means and that all member domains have the same interpretation of the common policy model (i.e., a common interpretation of a role-based access control model). Other work addresses specific aspects of bilateral authorization-check negotiation, for instance those that enable clients and servers to agree on common authorization properties; i.e., matching client credentials with server access checks [13,11]. Although this work on authorization-check negotiations introduces the important notion of client-server trust negotiation, it addresses neither the notion of access-management property negotiation (e.g., negotiation of common access distribution and revocation properties) in multiparty settings, such as those of dynamic coalitions and *ad-hoc* peer-to-peer networks, nor that of common state negotiation.

In contrast, our work explores the possibility of automating the negotiation of a common access state whenever all the domains of a coalition share the same interpretation of a common policy model. We cast this negotiation problem as one of satisfying diverse coalition-member objectives and use well-known results in game theory to find solutions. We also consider coalitions where member domains do not share a common policy model or model interpretation, and hence must negotiate them, since a common policy-model interpretation is a prerequisite for establishing common access states.

The rest of this paper is organized as follows. In Section 2 we present the negotiation of common states with examples. In Section 3 we discuss negotiation of policy model interpretations with some examples. Section 4 concludes the paper.

2 Negotiation of a Common Access State

We define the problem of negotiating a common access state as follows. We assume that a number of autonomous domains wish to collaborate to achieve a *common objective* that is desired by all domains and achievable by none of them individually. To achieve the common objective, these domains form a coalition and share some resources (e.g., data and applications). That is, each domain has a set of resources that are of interest to the others and is willing to share some or all of its resources with some or all of the other members to achieve the

common objective. Resource sharing by a domain with other domains means that the domain (administrator) grants access privileges for the resource being shared to the other domains. The sharing relationships among these domains are defined by the access state of the coalition. An access state is defined by the permissions each member domain has to the shared resources of that coalition. Thus negotiating a common access state means obtaining the agreement of each domain to share some of its data and applications with the other domains of the coalition. Such an agreement implies that each domain values achieving the common objective more than retaining private access to the resources it must share with other domains. In addition, a common access state of a coalition may include some jointly owned resources created during coalition operations, and a common policy for accessing these resources must be negotiated. Access to jointly owned objects is one of the benefits individual domains derive from their membership in the coalition.

In any coalition, there may be a number of common access states that satisfy the common objective. However, member domains may place some constraints on which of these states are acceptable; e.g., these constraints may arise due to technological limitations, or they may be self-imposed. For example, a domain whose participation is deemed essential to achieving the common objective might use its extra bargaining power to stipulate that it only accepts states in which every other domain contributes at least as many resources as it is contributing to the coalition. Even among the states it considers acceptable, a domain may prefer some states over others. For example, a domain may prefer to share certain resources only with those domains that it considers to be close allies; an example of such a situation can be found in [3]. The negotiation process aims to find a preferred common state that satisfies the constraints of all the domains. We expect that in a typical application these constraints would be programmed by the domain administrators at the start of the negotiation, and from that point on the process of finding a common state would be automated.

We divide negotiations into three classes based on the types of constraints under which domains negotiate. These classes, which are progressively generalized, differ from each other depending upon whether all the objectives of each domains (1) coincide with those of other domains and (2) are known to the other domains of the coalition.

1. **No constraints.** In this case all the objectives of all domains coincide and all objectives are known to all domains. In a typical case, all domains have a single common objective.
2. **Global constraints.** In this case some of the objectives of some domains may not coincide with those of other domains, yet all the domains have complete knowledge of each other's objectives.
3. **Local constraints.** In this case some of the objectives of some domains may not coincide with those of other domains, and domains may not have complete knowledge of each other's objectives.

Although some objectives of some domains may remain unknown, we assume that each domain has complete knowledge of which resources of interest

are possessed by the others. (An even richer set of negotiation problems would emerge, if we relax this assumption. That is, domains would have to guess at what resources other domains have, and tactics such as bluffing, commonly seen in real-life negotiations, would undoubtedly be used. These negotiation problems are outside the scope of this paper.) Given this assumption and the three negotiation classes, the following simple protocol arises naturally. The negotiation starts with one domain proposing a common state. Each of the other domains can then either accept this proposal or propose an alternative. Agreement is reached when all the domains accept a single common state. Within this simple negotiation protocol, the following four questions arise:

a. *Termination.* If a common state exists that satisfies all the constraints and all domains' objectives, will it be reached in the negotiation? How can the member domains detect if a negotiation is going to fail (if no mutually acceptable state exists)?
b. *Strategy.* What is the best negotiating strategy for each domain? In other words, how can a domain pick the content and/or sequencing of its proposals to ensure a more favorable outcome for itself?
c. *Preference specification.* How should domain preferences be specified to allow automated negotiation?
d. *Stability.* Is the negotiated state stable (i.e., self-enforcing)? In other words, is the common state such that no domain has an incentive to cheat or renege on its part of the agreement?

Below we illustrate the three classes of negotiations in some detail, and answer the above questions for each class.

2.1 Negotiation with No Constraints

In this type of negotiation, all domains have a common objective and are willing to accept any common state that satisfies that objective. Some of the previous work in this area [6,12] falls into this class; i.e., any proposal from the other domains is accepted, and extra-technological methods can be used to ensure in advance that only acceptable states will be proposed.

In this case, the answers to our four questions are obvious.

a. *Termination.* The negotiation always concludes successfully.
b. *Strategy.* The negotiation strategy is irrelevant, as any proposal will be accepted (usually, extra-technological agreements are made in advance to restrict abuse of this assurance by any domain).
c. *Preference specification.* Domain administrators need not specify anything beyond the common model of access policy and the common policy model interpretation.
d. *Stability.* Since the common objective is all-important, domains will not try to subvert the agreed-upon state. Hence, any state that helps to attain the common objective is stable.

As an example of this type of negotiation, consider a situation where a university needs to share some research data with engineers of some outside companies to get their feedback, with the *common objective* being the successful transfer of technology. In this case these companies simply tell the university which of their engineers are working on the relevant project, and the university role-based access control system places those engineers into local roles that have access to the necessary research data. Mechanisms for such role assignment have been studied by Shands *et al.* [12] and Herzberg *et al.* [6].

2.2 Negotiation with Global Constraints

Now consider the case where the coalition domains face (or have agreed upon) a set of constraints the common state must satisfy. Each domain knows all these constraints, and will accept any common state which satisfies all the constraints while achieving the common objective. It is possible that such a state does not exist, in which case the the negotiation fails. Note that since the constraints are known to all, any domain can compute the best common state independently.

For example, consider three domains representing three airlines that wish to share six route types to expand their market coverage. Here a route type corresponds to a certain set of destinations, for instance, U.S. - Europe, U.S. - Asia, U.S. - U.K.. Sharing a route implies that the domain which owns the route gives some users of a foreign domain the access privileges required to execute the route applications (e.g., reservations, billing, communications).

Table 1. Negotiation with global constraints: Route sharing.

Domain	Route Type Controlled	Number of routes in type
*	1	5
	4	8
1	5	4
	6	7
*	1	5
	2	6
2	3	2
	4	9
*	1	7
	2	4
3	4	6
	5	3

The initial state before negotiation is shown in Table 1. Observe that some route types may already be common to multiple domains; e.g., route types 1, 2, 4, and 5. Further, each route type consists of a number of routes, depending on which domain it belongs to. The *common objective* is to provide all three

domains with access to all six route types. The following global constraints have been agreed upon:

- Each domain must share at least one route type.
- Domains that have unique route types must share them. As a consequence, domain D1 must share route type 6 and domain D2 must share route type 3.
- Sharing of route types must minimize the number of routes shared (as possible application of the *least privilege principle*); i.e., if two or more domains are capable of sharing the same route type, then the one that comprises the lowest number of routes will be used.

Upon inspection of Table 1, we observe that two common states would satisfy these constraints. Domains D1, D2 and D3 could share route types {1,6}, {3} and {2,4,5} respectively (we call this Solution A). Or they could share {6}, {1,3} and {2,4,5} respectively (we call this Solution B). Either of these is acceptable to all domains, and as soon as one of them is proposed, it will be accepted. Any other state would violate at least one of the constraints, and will be rejected if proposed.

With this example in mind, let us try to answer our four questions.

a. *Termination.* Success or failure of negotiation depends on the specifics of the problem and on the constraints. However, each domain can always construct the set of common states that would be acceptable to all the others, and can predict whether the negotiation will fail or not by checking whether this set of acceptable states is empty.
b. *Strategy.* The domain that makes the first proposal has an advantage - it can construct the set of acceptable states and propose the one most advantageous to it, knowing that the proposal will be accepted. For instance, if D2 were to make the first proposal in the above example, it would propose Solution A, so that it has to share fewer of its own objects; similarly, if D1 were to make the first proposal, it would propose Solution B. Any other proposal would be meaningless, since the domain would know in advance that all the others would not accept it.
c. *Preference specification.* Domain administrators need to specify the information required to find the least-privilege solution, namely the data on which route type provides access to how many routes. In other words, data pertaining to the global constraint must be specified for the negotiation to be automated.
d. *Stability.* As long as the common objective is the primary goal for all the domains, no domain will try to subvert the agreed common state.

In the above example, we assumed that all the domains are committed to achieving the common objective. More complex negotiations may arise when domains have competing or even conflicting objectives. For example, consider the setting illustrated in Table 2. Here we have two are two domains, each of which controls a certain number of routes. Routes can be of three types. Each

domain has a set of three objectives it could pursue (e.g., markets to which the airline could cater), and each of these objectives has a certain value to that domain (e.g., the profit that could be made from that market). However, each such objective requires additional resources, in the form of routes, as shown in Tables 2(a) and 2(b).

Table 2. Game theory in negotiation

(a) Domain 1

Objective	Value of Objective	Type 1 Routes (U.S. - Europe) needed	Type 2 Routes (Europe - Asia) needed	Type 3 Routes (U.S. - Asia) needed
A	100	8	10	5
B	80	7	10	6
C	95	7	11	4

(b) Domain 2

Objective	Value of Objective	Type 1 Routes (U.S. - Europe) needed	Type 2 Routes (Europe - Asia) needed	Type 3 Routes (U.S. - Asia) needed
D	120	9	10	7
E	90	6	8	5
F	45	5	7	4

(c) Available resources

Domain	Type 1	Type 2	Type 3
1	20	20	11
2	18	19	11

We assume that each domain starts with a limited number of routes of each type (see Table 2(c)). The domains are unable to share routes, and can only trade some of their routes with each other in order to increase their profits. This is a route-trading (i.e., a barter) problem, unlike our previous example, which was a route-sharing problem.

In this case, the *common objective* is to increase profits for both domains through cooperation. However, this common objective no longer means the same thing to the two domains. The two domains will derive different profits from a successful negotiation, and each will seek to maximize its own profit. Profit maximization is an important aspect of most real-life negotiations, and we expect it to be a primary source of differences among the objectives of different domains.

Game theory [2] provides a mathematical framework for studying situations where individual rational decision makers are working towards competing or conflicting objectives. We can use game theory to find a solution for the problem in Table 2. First, observe that the best thing Domain 1 can do if it does not

collaborate with Domain 2 is to devote its resources to objectives A and B, and abandon objective C. This will result in a total profit of 180 to Domain 1. Similarly, Domain 2 should work towards objectives D and F, and abandon E, thus achieving a total profit of 165.

To find a barter arrangement that leaves both domains better off, we use the concept of the *nucleolus* [10]. The nucleolus is defined as the solution that minimizes the maximum dissatisfaction experienced by any one of the domains. The dissatisfaction is measured by the difference between what the domain could have achieved on its own and what it achieves as a result of cooperation. It can be shown that the nucleolus exists and is a stable outcome for very large classes of useful problems. Here, stability means that no single participant has both the power and the incentive to change the status quo. Thus, if any one participant in a negotiation tries to violate such an agreement, they will always be worse off than before, either because there is no more beneficial state for them or because the other participants can collaborate to punish the offender (for example by revoking his access privileges to their objects). A nice feature of the nucleolus in practice is that it is unique, and can usually be computed efficiently using linear programming techniques. If the nucleolus does not exist for a particular problem, its absence can also be detected efficiently.

For this example, the nucleolus solution is for Domain D1 to give one Type 3 route to Domain D2 in exchange for one Type 2 route. This will allow Domain D1 to fulfill objectives A and C, achieving a value of 195, while Domain D2 can fulfill objectives D and F, achieving a value of 210. Thus each domain achieves a higher value in this cooperative agreement, and therefore neither has any reason to try to violate it.

To illustrate the stability of the nucleolus in this case, we consider an alternative common state which also achieves the same profits for both domains. In this arrangement, Domain D1 gives two Type 3 routes to Domain D2 in exchange for one Type 2 route. However, Domain D1 can now depart from the agreement by withholding one of the two promised routes, without any ill effects (Domain D2 still achieves its best possible outcome, and so has no reason to try and penalize Domain D1). Thus this alternative is not a stable outcome. This observation is intuitively satisfying, since from a security point of view we can see that this alternative solution violates the least privilege principle (defined in the previous example).

In light of this second example, let us reexamine our four questions for the general case of any negotiation that can be cast in terms of objective values and resources, as in Table 2. (Note that our previous example (in Table 1) can also be recast in this form.)

a. *Termination.* All domains can check if the negotiation will succeed by checking if a nucleolus exists, if it does the negotiation is guaranteed to succeed.
b. *Strategy.* The best negotiating strategy is to propose the nucleolus.
c. *Preference specification.* Negotiation can be automated if domains specify the different objectives, their values, and resource limitations.

d. *Stability.* The nature of the nucleolus ensures that no domain will try to subvert it, since it is now in their interest to help achieve the common objective.

2.3 Negotiation under Local Constraints

Suppose, as in the previous section, that a number of domains with a common objective are seeking to negotiate a common access state. Each domain has its own constraints and preferences, as well as its own estimate of the importance of the common objective; i.e., each domain has its own objectives that may differ from those of other domains. It will accept any common state that satisfies all its constraints, fulfills the common objective, and is preferable to the outcome of non-cooperation. In other words, a domain will not agree to accept a common access state that requires it to give up more than it can gain from a negotiation that achieves the common objective.

Now suppose that no domain has complete knowledge of the constraints and preferences of all the other domains. Thus each domain must try and negotiate the best state for itself without knowing precisely what resources the other domains are able (or willing) to contribute to the common access state. Negotiation strategies become very important, since a domain cannot determine beforehand what is the best common state (as could be done in the case of negotiations with global constraints). Also, a domain cannot predict whether a common state exists that satisfies all the constraints of all the domains, so a domain cannot predict whether the negotiation will succeed. Detecting success or failure becomes part of the negotiating strategy, and a decision as to whether to continue negotiating must be made at each step of the negotiation. For example, consider the problem in Table 2, and assume that the contents of Table 2(a) are known only to Domain 1 and the contents of Table 2(b) are known only to Domain 2. This would constitute a negotiation with local constraints.

Another example of negotiating under both local constraints is shown in Table 3. This is a route sharing problem, with three domains and twelve route types. The *common objective* is to increase profits for all domains by increasing the number of routes accessible to each domain. Each domain has some constraints which it can reveal to the other domains, and some which it cannot. Note that if all constraints could be revealed, this would become a negotiation with global constraints. The local constraints imposed by each of the three domains are as follows:

- Domain D1: willing to share half as many routes as other domains because of some (e.g., geographic) advantage. D1 can reveal this constraint.
- Domain D2: willing to share twice as many routes as D1; unwilling to share route 6 and cannot reveal the constraint regarding route 6.
- Domain D3: willing to share twice as many routes as D1; needs one of the routes {2, 5, 6}, is unwilling to share route 11 and cannot reveal this constraint regarding route 11.

One way to carry out the negotiation would be as follows: all the domains construct lists ranking the various feasible common states in their order of prefer-

Table 3. Negotiation under local constraints

Domain	Routes Controlled	Routes shared in final solution
D1	1,2,3	2
D2	4,5,6,7,8	4,5
D3	9,10,11,12	9,10

ence, based on their local constraints. Each domain starts by proposing its most preferred state, and proceeds down the list if this proposal is not accepted. If the number of feasible common states is finite (as in this example) this negotiation will terminate, with success if some agreement is reached or with failure if all domains reach the end of their lists without obtaining agreement. The following two examples show negotiations that terminate in agreement.

1. – D1 proposes route-sharing state {D1: 1, D2: (4, 5), D3: (9, 10)}
   – Result: all domains agree and common state is committed
2. – D3 proposes route-sharing state: {D1: 2, D2: (5, 6), D3: (9, 10)}
   – Domain Response: D2 rejects it (unwilling to share 6), and instead proposes: {D1: 2, D2: (4, 5), D3: (10, 11)}
   – Domain Response: D3 rejects it (unwilling to share 11) and instead proposes: {D1: 2, D2: (4, 5), D3: (9, 10)}
   – Result: all domains agree and common state is committed

However, this approach has a problem, as shown by Arrow [1]. The Arrow Impossibility Result states that if each domain describes their preferences by a preference list (a preference list is a weak order, where an alternative A appears above another alternative B only if the entity making the list considers A to be at least as good as B), and if an impartial arbitrator tries to use these lists to construct an overall preference list for all the domains, then there is no algorithm for the arbitrator to always construct a list that is

– Pareto efficient; i.e., there is no other list that would make at least one domain better off (by ranking its more preferred alternatives higher) without making some other domain worse off.
– Nondictatorial; i.e., there is no one domain whose preferences are always followed, regardless of the other domains' preferences.
– Independent of irrelevant alternatives; i.e., the decision to rank alternative A above or below alternative B is taken only on the basis of different domains' preferences between A and B, and is not affected by whether, for instance, some domain ranks a third alternative C above or below A or B.

Furthermore, it can be shown that if the arbitrator's decision is restricted to just picking a most preferred solution instead of constructing a full preference list, then there is no algorithm for the arbitrator which is nondictatorial and

non-manipulable (i.e., some domain can achieve a better result by lying about its preferences).

This result has strong implications when we seek to answer our fourth question. It implies that a negotiation scheme based on preference lists will either not conclude, or if it does, the common state reached may not be stable, because

- it may not be Pareto-efficient, in which case there might be another common state that is better for all the domains.
- it may be dictatorial, in which case it is unfair to all but the "dictator" domain. This may cause the other domains to try and subvert the outcome of the negotiation.
- it may not be independent of irrelevant alternatives, in which case any domain could influence the outcome of the negotiation by proposing some extra common states, even infeasible ones.

Thus, Arrow's result suggests that, to construct useful solutions to such negotiation problems, we must at least have a way of defining *cardinal preferences*. In other words, it is not enough for each domain to be able to rank the possible outcomes; it should be able to quantify the strength of its preference for each outcome. Thus the information required to assign a numeric value to any given common state must be specified by the system administrator. This was the case in Table 2, where preferences were quantified as the value associated with each objective.

This discussion and attempts to provide answers to our four questions for the case of negotiation with local constraints opens several avenues of further research:

a. *Termination.* This question is strongly linked to the negotiating strategies adopted by the domains. Finding negotiation schemes and strategies that attain stable outcomes in an efficient way while allowing domains to detect failing negotiations is an interesting area for future research.
b. *Strategy.* It is not clear what strategies will be the best in this case. We conjecture that the best strategy would be for each domain to start with an initial offer and then make concessions if the other domains show a willingness to compromise, or break off the negotiations if the other domains are adamant. Known results of game theory [2] also point in this direction, but this has not been proven yet.
c. *Preference specification.* Arrow's result indicates that each domain must at least specify the strength of the domain's preference for all possible common states, or provide a rule for computing the strength of the preference given a common state.
d. *Stability.* This question is also related to the negotiating strategy and remains an area of future research. For example, if we could find a strategy that allows the domains to converge to the nucleolus in a finite number of negotiating rounds, then such a solution would always be stable.

3 Negotiation of Policy Models

In this section we discuss cases when negotiation of common policy models and common policy-model interpretations becomes necessary. We first describe the three types of policy properties that comprise a policy model. Then we give examples of negotiation of these properties in trust negotiation [13] and in dynamic coalitions.

Negotiating policy models involves negotiation of policy properties, namely, *access-attribute* (AT), *access-management* (AM) and *access-authorization* (AA) properties [4,5]. Access attributes include subject and object attributes (e.g., user, group, role, location identifiers, secrecy and integrity levels, time-of-access intervals), and AT properties typically establish invariant relationships among attributes (e.g., lattices of secrecy and integrity levels, user-group membership invariants, inheritance of role membership and/or role permissions). AA properties determine whether a subject's current access attributes satisfy the conditions for accessing an object given the current object attributes. AM properties include conditions under which subjects are granted or revoked attributes for accessing objects, subjects and objects are created or destroyed, objects are encapsulated within protected subsystems. In general, AM properties characterize commands and constraints that bring the system to desired states, whereas AA properties characterize constraints that ensure that the system remains in desired states.

A negotiation of AA and AT properties can be identified in trust negotiations between a client and a server [11,13]. The client sends requests to the server and the server needs to verify the client's credentials in order to authorize the request. However, the server is unwilling to disclose up front the AA rules; i.e., the type of credentials that will allow the client to have his request approved. The purpose of this is to keep some privileged association private, for example, if the company to which the server belongs is in an alliance with another company C and wishes to keep this fact private, then disclosure of the fact that a certificate from company C will allow the requested service would result in exposing the alliance. Also, the client is reluctant to disclose all his credentials along with the initial request in order to keep some privileged associations private and to protect sensitive private information (such as social security number, bank account number). So the client and server engage in a negotiation process where clients must provide credentials whose AT properties can be verified by the server, and the server discloses further AA rules to enable the client's request to be satisfied (the client may also require the server to present some credentials whose AT properties can be verified by the client).

A need for negotiating AM properties can be seen in a case when two domains, both of which use role-based access control models, are trying to share resources but each domain maintains different "rules" for assigning users to roles. Then the domains need to engage in a negotiation for a common set of rules that each one is going to use to assign foreign users to its local roles. For example, let Domain1 and Domain2 represent two airlines. Domain1 requires a user to present an identity certificate and a digitally signed proof of job title, in order to assign him to the local role of Travel_Agent1. For the assignment to a similar local

role of Travel_Agent2, Domain2 requires a user to present an identity certificate and a conventional company-identification credential. So the domains need to negotiate either one of the existing sets of credentials or a new set of credentials for foreign user assignment to roles. For example, Domain2 can agree to accept digitally signed job titles or alternatively, or both domains may agree to accept digitally signed credentials issued by a trusted third party.

Another scenario for negotiation of AM properties is possible in regard to selective revocation of attribute certificates [7,8]. Selective revocation means that whenever an identity certificate is revoked, so are all the attribute certificates that were issued based on this identity certificate. Now consider the situation when two domains are trying to collaborate, but one of them supports selective revocation and the other one does not. The domains must then negotiate AM properties of attribute certificate revocation before distributing privileges to users. There are three options, each of which may cause a change in AA properties. The first option is that both domains agree to support selective revocation, then on every access request only the attribute certificate's validity need to be checked. The second option is that both domains agree not to support selective revocation. In such a case both identity and access certificates need be verified. The third option is to have each domain support its individual AM properties. In such a case it is necessary to distinguish requests from users from different domains and for each case use corresponding authorization policies.

4 Summary and Further Research

We defined the problem of negotiation of access control policies in the general setting of dynamic coalitions and *ad-hoc* peer-to-peer networks. We presented examples of negotiating common access states and common policy-interpretation properties, and identified some areas of future research. We intend to develop a prototype for establishing dynamic coalitions among autonomous domains and for conducting negotiation of common access states under different types of constraints. This prototype will include a flexible language for negotiation that can capture a large set of real-life examples, not just the examples mentioned in this work. We also intend to illustrate how game theory can be applied to negotiations in practice, for example in negotiations where cardinal preferences can be defined.

Acknowledgments. This work is supported by the Defense Advanced Research Projects Agency and managed by the U.S. Air Force Research Laboratory under contract F30602-00-2-0510. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, U.S. Air Force Research Laboratory, or the United States Government.

References

[1] K. J. Arrow. *Social Choice and Individual Values.* Yale University Press, New Haven, CT, second edition, 1963.

[2] F. Forgo, J. Szep, and F. Szidarovsky. *Introduction to the Theory of Games: Concepts, Methods, Applications*, volume 32 of *Nonconvex Optimization and Its Applications.* Kluwer Academic Publishers, 1999.

[3] T. J. Gibson. An architecture for flexible multi-security domain networks. In *Proceedings of Network and Distributed System Security Symposium*, San Diego, CA, February 2001. The Internet Society.

[4] V. D. Gligor, S.I. Gavrila, and D. Ferraiolo. On the formal definition of separation-of-duty policies and their composition. In *Proceedings of the 1998 IEEE Symposium on Security and Privacy*, Oakland, California, May 1998.

[5] V.D. Gligor and S.I. Gavrila. Application-oriented security policies and their composition. In *Proceedings of Security Protocols 6th International Workshop*, Cambridge, UK, April 1998.

[6] A. Herzberg, Y. Mass, J. Michaeli, D. Naor, and Y. Ravid. Access control meets public key infrastructure, or: Assigning roles to strangers. In *Proceedings of IEEE Symposium on Security and Privacy*, Oakland, California, May 2000.

[7] H. Khurana and V.D. Gligor. Review and revocation of access privileges distributed with PKI certificates. In *Proceedings of the 8th International Workshop on Security Protocols*, volume 2133, pages 100–125, Cambridge, UK, April 2000.

[8] H. Khurana and V.D. Gligor. Enforcing of certificate dependencies in ad-hoc networks. In *Proceedings of the IEEE International Conference on Telecommunications*, Romania, April 2001. ISBN: 973-99995-1-4.

[9] H. Khurana, V.D. Gligor, and J. Linn. Reasoning about joint administration of access policies for coalition resources. Submitted for publication. Available at http://www.glue.umd.edu/~gligor.

[10] D. Schmeidler. The nucleolus of a characteristic function game. *SIAM Journal on Applied Mathematics*, 17:1163–1170, 1968.

[11] K. E. Seamons, M. Winslett, and T. Yu. Limiting the disclosure of access control policies during automated trust negotiation. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, California, February 2001.

[12] D. Shands, R. Yee, J. Jacobs, and E. J. Sebes. Secure virtual enclaves: Supporting coalition use of distributed application technologies. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, California, 3-4 February 2000.

[13] W. H. Winsborough, K. E. Seamons, and V. E. Jones. Automated trust negotiation. In *DARPA Information Survivability Conference and Exposition (DISCEX 2000)*, Hilton Head, SC, January 2000.