RT-Based Administrative Models for Community Cyber Security Information Sharing

Ravi Sandhu*, Khalid Zaman Bijon*, Xin Jin*, and Ram Krishnan†
*Institute for Cyber Security & Department of Computer Science
†Institute for Cyber Security & Department of Electrical and Computer Engineering
{ravi.sandhu, ram.krishnan}@utsa.edu , {kbijon, xjin}@cs.utsa.edu
University of Texas at San Antonio

***Abstract*—We develop a series of formal administrative models for recently proposed informal requirements for community cyber security information sharing [9]. Traditional enterprise-oriented administrative models are not suitable for the highly dynamic and distributed nature of this multi-organization application domain. Hence, new administrative models with robust intuitive grounding and rigorous mathematical foundations are required. We show that the role-based trust management (RT) framework [5], [7] is suitable in large measure to address the informal scenarios of [9], with one essential extension to enable self-assignment of users to selected roles. Applications of extended forms of RT, as well as its limitations, are also considered.**

Keywords: Community Cyber Security, Trust Management, Secure Information Sharing.

I. Introduction

Effective and secure information sharing, especially across multiple cooperating yet mutually suspicious organizations, is a fundamental challenge in today's information-rich and information-dependent society. The necessity to *share but protect* is among the oldest challenges for trustworthy computing. Many of the traditional approaches have focussed on a single enterprise which does not generalize easily to multiple organizations. For the past decade, in the US there has been a persistent call for effective information sharing for homeland security. Much effort has been placed on national level information sharing and analysis centers across various industry and government sectors. In this paper we look at the other extreme of secure information sharing at the community level for the purpose of cooperating on cyber security incidents that impact more than one organization. Community in this context refers to a county or larger city size entity with a well demarcated geographical boundary closely aligned with a governance boundary. Currently there is no widely accepted set of informal requirements, let alone formal models, for supporting community cyber security information sharing. An initial informal set of requirements was recently proposed [9].

Our central contribution in this paper is to develop formal administrative models for these recently proposed informal scenarios. For this purpose we use the well-known role-based trust management (RT) framework [5], [7], which to our knowledge has not been previously applied in this domain. Our motivations for choosing RT include its strong mathematical foundations, efficient safety analysis, explicit inclusion of roles and its sizable literature. Our principal finding is that RT is suitable in large measure to address the informal scenarios of [9], with one essential extension to enable self-assignment of users to selected roles. We also show how extended forms of RT can be useful in this context, as well as discuss limitations of the RT approach revealed by this exercise.

The remainder of this paper is organized as follows. Section II reviews the previously published community cyber security information sharing informal requirements and the role based trust management framework. Section III develops a series of formal RT_0-based administrative models for these informal scenarios. Section IV presents additional features of these scenarios and expresses them using extended forms of RT, beyond RT_0. Section V discusses the limitations of the RT approach and section VI gives our conclusions.

II. Background

A. Community Cyber Security Information Sharing

Recently, Sandhu et al [9] proposed a methodology and informal requirements for information sharing for cooperative cyber incident management in a community. These requirements were abstracted from the decade long experience of the Center for Infrastructure Assurance and Security (CIAS) at the University of Texas at San Antonio. Over the past decade CIAS has conducted cyber security preparedness exercises and training at communities throughout the nation specifically dealing with communication, incident response, disaster recovery, business continuity, security awareness and similar issues. Among the fundamental requirements is the need to accommodate multiple organizational administrative domains and the need to rapidly assemble and enable dynamic cross-organizational incident teams.

These requirements were found to be a good match for the concept of group-centric secure information sharing (g-SIS) [2], [3], [4]. The motivating metaphor of g-SIS is that of a group as a secure virtual meeting room where participants and information are brought together to share for some collaborative common purpose. A variety of policies dealing with the temporal aspects of users and objects entering and departing from the "room" have been formalized and their security properties thoroughly analyzed. The administrative aspects of authorizing entry and departure have been less

COLLABORATECOM 2011, October 15-18, Orlando, United States

DOI 10.4108/icst.collaboratecom.2011.247176

studied so far. Groups in g-SIS models are classified as isolated or connected. Isolated groups do not interact with each other whereas connected groups have interactions and dependencies. For instance, a user's membership in one isolated group has no implication on her authorizations in other groups. Likewise an object's availability to one isolated group has no dependence on availability in a different group.

No single administrative model for g-SIS can possibly apply to all the scenarios where g-SIS is relevant. An example of a formal administrative policy for a single collaboration group between two organizations is given in [4]. Thus far there has been no formal administrative model for connected groups.

The informal sharing and administrative requirements developed in [9] are illustrated in figure 1 and 2 with slight adaptation. Figure 1 illustrates two long-lived steady state groups labelled as core and open, and a transient incident group assembled to respond to a specific cyber incident. These three groups are characterized as follows.

- Core Group: In the steady state, one long-lived core group is formed. Information in this group is highly sensitive and is restricted to be shared between core group members. Membership in the core group is tightly controlled and is limited to members of relevant organizations in the community. Further details on how this membership should be administered are not specified in [9].
- Open Group: We would also have one open group in the steady state. This group enables sharing of relatively insensitive and even publicly available information. Membership in this group is voluntary and available to a sizable subset of the total population of members of relevant organizations in the community. Moreover, users in the core group are automatically enrolled in the larger open group so they can participate and observe. Additional details which would be required to develop a formal model are not specified in [9].
- Incident Group: In response to an incident, an incident group is established by administratively designating a selected set of users and objects from the core group. A conditional membership relationship is established between the core and incident groups ensuring membership of such users in the incident group is contingent upon their continued membership in the core. Selected members of the open group may also be brought into the incident group, for instance based on their reputation or expertise with respect to aspects of the incident. Further, domain experts from outside the community organizations may be recruited to this group. Thus, an incident group has a mix of members: some who are core group members, others who are open group members and yet others who are domain experts external to the community organizations. The filtered read and filtered read-write arrows in figure 1 enable incident group members to have selective access to information objects in the core and open groups respectively. Multiple incident groups may coexist at the same time as different incidents occur, as shown in figure 2. Once the incident has been closed its incident group may be discarded and possibly archived. Here again additional details are required beyond those outlined in [9].

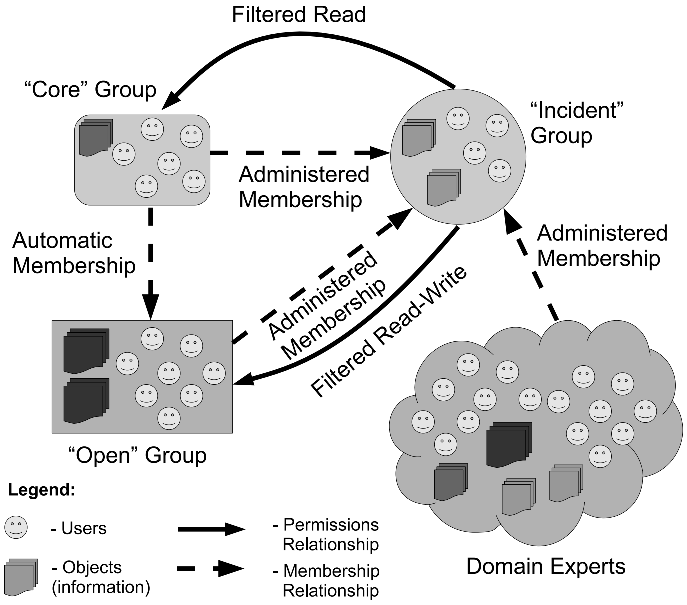

Fig. 1. *Life cycle of a typical community cyber incident.*

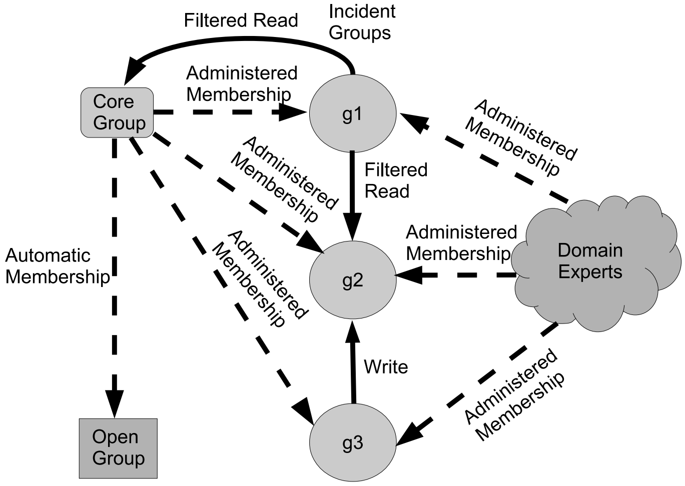

Fig. 2. *Dynamic community cyber incident management.*

These three groups are shown in figures 1 and 2 using different icons (rounded rectangle for the core group, rectangle for the open group and circles for the incident groups) to emphasize that they have different characteristics. The domain experts are indicated by yet another icon, a fuzzy cloud rather than a sharply defined icon of regular geometry. This covers experts from external entities, such as university researchers or private consultants, possibly beyond the local community. In g-SIS this collection is not a group since there is no information sharing amongst the domain experts via this means. It is simply a collection of external users who may be admitted to an incident group but not to the open or core groups.

To summarize, figure 1 informally illustrates the application of connected groups to community cyber security information sharing. Each of the core, open and incident groups has different sets of users and objects (information), and different administrative procedures for admitting members. The dashed single-headed arrows indicate different types of conditional membership relationships between groups. For example, the automatic membership relationship between the core and open groups indicate that members in the core group get automatic membership in the open group so long as they remain members in the core group. As another example, the administered membership between the core and incident groups indicate that some core group members are also made members of the incident group. Again, their membership in the incident group is contingent upon their membership in the core group.

The solid arrows indicate the permissions that users in one group may exercise on another in the direction of the arrow. As indicated in figure 2, there may be multiple incident groups that simultaneously co-exist, including the capability for one incident group to access or inject information into another.

B. Role Based Trust Management (RT)

RT is a family of role-based trust management languages introduced by Li et al [5], [7]. At its most abstract, the notion of role used is simply a set of principals. The primary application of RT is intended to be authorization and access control. An RT role enables its members to access specific resources assigned to that role. The assignment of resources to roles is similarly modeled in RT, as described later.

In this paper we will primarily use RT_0 which is the base and main member of the RT family. The basic constructs of RT_0 are entities, role names and roles defined as follows.

- Entities: Entities are also often called principals. They can define roles, issue credentials, and make requests. They are denoted by names starting with an uppercase letter (possibly with a subscript), e.g. *A*, *B*, B_1, and *Alice* are all entities.
- Role names: Role names are denoted by strings starting with a lowercase letter (possibly with a subscript), e.g. *r*, r_1, and *student*.
- Roles: Roles have the form of an entity followed by a role name, separated by a dot, e.g. *A.r*, $B.r_1$, and *University.student* are valid roles.

Permissions in RT_0 are represented by roles. For instance, the permission to read confidential document on a corporate network of a company C can be represented by role *C.readConfidential*. Here, an entity has the read permission if and only if it belongs to *C.readConfidential*. By aggregating these permissions, roles can represent sets of permissions.

There are four types of credentials in RT_0 that an entity *A* can issue, each corresponding to a different way of defining the membership of one of *A*'s roles *A.r*.

$$\textbf{Simple Member:} \quad A.r \leftarrow D \tag{1}$$

A asserts that *D* is a member of *A.r*.

$$\textbf{Simple Inclusion:} \quad A.r \leftarrow B.r_1 \tag{2}$$

A asserts that *A.r* includes (all members of) $B.r_1$.

$$\textbf{Linking Inclusion:} \quad A.r \leftarrow A.r_1.r_2 \tag{3}$$

$A.r_1.r_2$ is called a linked role. By issuing this credential *A* asserts that *A.r* includes all members of $B.r_2$ for every *B* that is a member of $A.r_1$.

$$\textbf{Intersection Inclusion:} \quad A.r \leftarrow B_1.r_1 \bigcap B_2.r_2 \tag{4}$$

A asserts that *A.r* includes every principal who is a member of both $B_1.r_1$ and $B_2.r_2$.

There are five additional components of the RT framework: RT_1, RT_2, RT^T, RT^D and $RT_\ominus$. Each of them add different features to RT_0. RT_1 adds parameterized roles to RT_0, which can express attribute fields. RT_2 adds logical objects to RT_1, which can group logically related objects together so that permissions about them can be assigned together. RT^T provides manifold roles and role-product operators, which can express threshold and separation-of-duty policies. RT^D provides delegation of role activations, which can express selective use and delegation of privileges. $RT_\ominus$ deals with restricted form of negation. Their application to our scenario is discussed in section IV.

Required Enhancement of RT: The core administration model of RT_0, and its extensions, is defined as follows [**?**] without further elaboration.

> "The entity the role belongs to is called the owner of the role, and is the only authority that can directly determine which are the members of the role."

To accommodate our scenario we will need to extend this rule to allow self-assignment to designated roles (section III).

Operational Limitation of RT: It should also be noted that RT does not provide specific operational statements for removing principals or permissions from a role. RT requires that the entire policy is available as a collection of RT statements. Any changes in policy require a complete substitution of a new policy for the old. As such RT is not an operational language for incrementally adjusting policy, including simple and frequent adjustments such as revoking membership of a specific user who has left the organization. Operationalizing RT in this sense would be required to actually deploy RT based systems in practice.

C. Other Related Work

Several administrative models have been proposed for various contexts, including the following. ARBAC97 [8] specifies a comprehensive administrative model for RBAC [10]. It assumes that there is a set of administrative roles, *AR*, which is disjoint from the set of normal roles. Only members of administrative roles can perform administrative operations. Bhatti et al [1] specify a fine-grained policy for integration of federated identity and privilege management. Their policy integrates a decentralized single sign-on mechanism within an authorization model by adapting it to use property based Trust Management credentials. The Secure Virtual Enclaves

[11] collaboration infrastructure allows multiple organizations to share their distributed application objects, while retaining organizational autonomy over local resources. A more comprehensive case study for Community Cyber Security Information Sharing could include elements of these models.

III. RT-Based Administrative Models

As discussed earlier, the core, open and incident groups have different sets of users and resources. In general there is some degree of overlapping users and objects across groups, while the purpose and membership criteria for the groups varies. In the core group, the participating local organizations nominate their own representatives while an individual can voluntarily join the open group if she possesses some enabling credentials. One of the main challenges in building an administrative model in this context is to manage user membership in these groups given a large population of users whose credentials are decentralized amongst various organizations in the community. Moreover, users may join or leave the organizations or groups unpredictably. In this section we develop a series of RT_0-based formal models [**?**] to demonstrate how this process can be automated and decentralized, at various levels of sophistication, through the RT mechanism of credential chains. Thereby we fill in details not explicitly addressed in the informal model.

We treat these three kinds of groups in turn as follows.

A. Membership Management in the Core Group

Recall that there is a single long-lived core group regardless of whether or not there are ongoing incidents. We represent the core group as an entity *CG*. We use the role name *user* so that the role *CG.user* denotes all users who are members of the core group. The actual membership is determined by the individual organizations in the community. To be concrete we will consider the city of San Antonio as the hypothetical community in question, and use the entity *SAT* to represent this community. Further we use the role name *member*, so that the role *SAT.member* comprises organizations that are members of the San Antonio community. Finally, we use 3 organizations in our example represented as the entities *CPS*, *SAWS* and *SAPD* which are respectively San Antonio's energy utility, water and police departments. The RT_0 statements below express this.

$$\mathit{SAT.member} \leftarrow \mathit{CPS} \tag{5}$$

$$\mathit{SAT.member} \leftarrow \mathit{SAWS} \tag{6}$$

$$\mathit{SAT.member} \leftarrow \mathit{SAPD} \tag{7}$$

Obviously we could add additional organizations in this manner under control of the entity *SAT*. Authority to assign users to *CG* is delegated to these individual organizations by the following statement which can be specified only by *CG*.

$$\mathit{CG.user} \leftarrow \mathit{SAT.member.cgrep} \tag{8}$$

In other words members of the *X.cgrep* role where *X* is an organization that is a member of *SAT* will constitute the users of *CG*. Each individual organization assigns human users to its *cgrep* role by the following statements, for example.

$$\mathit{CPS.cgrep} \leftarrow \mathit{Alice} \tag{9}$$

$$\mathit{CPS.cgrep} \leftarrow \mathit{Bob} \tag{10}$$

$$\mathit{SAWS.cgrep} \leftarrow \mathit{Carol} \tag{11}$$

$$\mathit{SAPD.cgrep} \leftarrow \mathit{Dan} \tag{12}$$

Thereby via the credential chains *Alice*, *Bob*, *Carol* and *Dan* are members of the *CG.user* role. A San Antonio size community (population 1.3 million in 2010 census) would have a few tens of organizations with each having one or a couple of representatives in the core group. Thus the above is an appropriate solution for managing core group membership. The entity *CG* could be managed by an appropriate cross-organization entity such as the Office of Emergency Management that typically exists in most community.

B. Membership Management in Open Group

Let us represent the open group as the entity *OG* with the role *OG.user* representing its members. The following statement makes all core group members automatically also members of the open group.

$$\mathit{OG.user} \leftarrow \mathit{CG.user} \tag{13}$$

Next consider how to permit a suitable population of users from *SAT.member* organizations to become a *OG.user* at their own volition. One possibility is to open up membership to all employees which could be represented by *SAT.member.employee*. A more restricted population would be represented by the role *SAT.member.itmember*. Each organization gets to choose who should be an *itmember* in that organization, such as the following.

$$\mathit{CPS.itmember} \leftarrow \mathit{Eve} \tag{14}$$

$$\mathit{CPS.itmember} \leftarrow \mathit{Fred} \tag{15}$$

$$\mathit{SAPD.itmember} \leftarrow \mathit{Gary} \tag{16}$$

The following statement would then automatically make every *itmember* of every *SAT.member* organization a member of OG.

$$\mathit{OG.user} \leftarrow \mathit{SAT.member.itmember} \tag{17}$$

However, this does not give *Eve*, *Fred* or *Gary* any say in the matter. The core RT administrative policy is that the entity *OG* controls membership in the role *OG.user*. To accommodate the volition of *Eve*, *Fred* and *Gary* in joining *OG.user* we need to extend the core RT administrative model in a small but essential aspect. We introduce the concept of an **open role** represented by an underlined role name as in $\mathit{OG.}\underline{\mathit{volunteer}}$. For simplicity, we require each role name has to be declared to be open or not-open, so only one of the role names *volunteer* and $\underline{\mathit{volunteer}}$ can occur in an RT policy.

$$\mathit{OG.}\underline{\mathit{volunteer}} \leftarrow \mathit{Eve} \tag{18}$$

The crucial point is that this operation occurs at *Eve*'s discretion, not *OG*'s discretion. The following statement then captures the informal policy requirements for *OG* membership.

$$\mathit{OG.user} \leftarrow \mathit{SAT.member.itmember} \cap \mathit{OG.}\underline{\mathit{volunteer}} \tag{19}$$

This statement will replace statement (17) above and will enable *Eve*'s membership in *OG.user*.

One might ask what is the difference in *Eve* joining a group voluntarily via (18) and (19), versus being automatically assigned to the group and simply ignoring the group via (17)? If nothing else the former enables *Eve* to prevent herself from being volunteered for activities she does not wish to undertake. Volunteering to join *OG* can have a cascading effect. Looking ahead to (21), *Eve*'s membership in *OG.user* enables *IG* to recruit her into *IG.user* regardless of her level of activity in *OG*. Now she further has to be inactive in *IG* to assert her lack of interest in *OG* and related groups. In other words, voluntarily joining a group denotes commitment whereas lack of activity in a group does not preclude assignment to other related groups. Moreover, lack of activity in a group will likely reflect very differently on reputation of the individual versus lack of volunteering to join a group.

C. Membership Management in Incident Group

An incident group differs from core and open groups in several respects. It is transient rather than permanent and there may be several instances at any time. The nature and duration of an incident will drive the group's life cycle and membership.

Single Incident Group: Let us first consider a single incident group *IG* with *IG.user* representing its members. We require a two part authorization to become a member of *IG*. One part, under *IG* control is represented by the following statements.

$$\mathit{IG.authorized} \leftarrow \mathit{Alice} \tag{20}$$

$$\mathit{IG.authorized} \leftarrow \mathit{Eve} \tag{21}$$

This represents authorization by *IG* for *Alice* and *Eve* to join *IG*. The requirement that membership in *IG* is conditional on membership in *CG* or *OG*, is expressed below.

$$\mathit{IG.user} \leftarrow \mathit{CG.user} \cap \mathit{IG.authorized} \tag{22}$$

$$\mathit{IG.user} \leftarrow \mathit{OG.user} \cap \mathit{IG.authorized} \tag{23}$$

Recall that *Alice* and *Eve* are respectively members of *CG.user* and *OG.user* respectively due to the RT_0 statements in sections A and B respectively. So the above statements make both of them members of *IG.user*. To account for domain experts we assume there is a *SAT.domainexpert* role to which a pool of domain experts are assigned under control of *SAT*. The following statements then make *Hilda* a member of *IG.user*.

$$\mathit{SAT.domainexpert} \leftarrow \mathit{Hilda} \tag{24}$$

$$\mathit{IG.authorized} \leftarrow \mathit{Hilda} \tag{25}$$

$$\mathit{IG.user} \leftarrow \mathit{SAT.domainexpert} \cap \mathit{IG.authorized} \tag{26}$$

Thereby *Alice*, *Eve* and *Hilda* become members of *IG.user* conditional on continued membership in *CG.user*, *OG.user* and *SAT.domainexpert* respectively. Next consider the requirement that *IG* members have filtered read access to documents in *CG* and filtered read-write access to documents in *OG*. For this purpose, we construct a role *CG.filtered-read* to which read access to selected documents of *CG* is provided. Likewise there is a role *CG.filtered-read-write* to which read-write access to selected documents of *OG* is provided. The following statements then enable filtered access by *IG* members to a subset of documents in *CG* and *OG*.

$$\mathit{CG.filtered\text{-}read} \leftarrow \mathit{IG.user} \tag{27}$$

$$\mathit{OG.filtered\text{-}read\text{-}write} \leftarrow \mathit{IG.user} \tag{28}$$

Multiple Incident Groups: Now consider multiple incident groups as shown in figure 2 so we have corresponding entities IG_1, IG_2 and IG_3. The filtered read relationship from IG_1 to IG_2 can be accommodated as above. For the write access we assume there is a role $IG_2.write$ which authorizes write access in a suitable manner. The following RT_0 statements then enable the required authorizations.

$$IG_2.\mathit{filtered\text{-}read} \leftarrow IG_1.\mathit{user} \tag{29}$$

$$IG_2.\mathit{write} \leftarrow IG_3.\mathit{user} \tag{30}$$

IV. Extended Features

In this section we consider some enhancements to the previously considered scenarios so as to demonstrate the capabilities of extended forms of RT in this context.

Single Document Release to Incident Group: Suppose that the incident group *IG* requires a single document from *CG*. One solution might be to add this document to the *CG.filtered-read* role. However, the *CG.filtered-read* role may be used for multiple incident groups to provide identical filtered-read to each one. Thus, it would be desirable to have a mechanism for release of a single document to a single incident group. From the previous RT statements we know that *Alice*, *Bob*, *Carol* and *Dan* are members of the *CG.user* role, and *Alice* is also a member of *IG.user* role. Let us say that the release policy stipulates that a request for the document must come from an *IG.user* and must be approved by a *CG.user* who is not also an *IG.user*. The latter part of this sentence imposes a separation of duty requirement, so *Alice* cannot approve release of the document to *IG.user*. Release approval must come from someone else such as *Bob*, *Carol* or *Dan*. This requirement can be expressed using RT_1 and $RT_\ominus$ as follows.

$$\mathit{CG.read(?o)} \leftarrow \mathit{CG.permit(?o)} \cap \mathit{IG.user} \tag{31}$$

$$\mathit{CG.permit(?o)} \leftarrow \mathit{CG.approve(?o)} \cap \mathit{IG.request(?o)} \tag{32}$$

$$\mathit{CG.approve(?o)} \leftarrow \mathit{CG.user.approve(?o)} \ominus \mathit{IG.user} \tag{33}$$

These statements apply to a single object represented as *?o* as per the parameterized roles of RT_1. The symbol $\ominus$ represents mutual exclusion. These statements assert that a specific *CG* object can be read by a member of *IG.user* provided the object was requested by some *IG.user* and its release approved by some *CG.user* who is not also an *IG.user*. Thus in our example *Alice* can make the request but cannot approve it.

Delegation of Role Activation: Delegation of role activation is another desirable feature. For instance, consider *Alice* who is a member of *IG.user* as well as *CG.user*. If she is unavailable for some reason it may be necessary to substitute her with say Bob in *IG*. The following statement from RT^D achieves this.

$$\mathit{Bob} \xrightarrow{\mathit{Alice}\quad \mathit{as}\quad \mathit{IG.user}} \mathit{access} \tag{34}$$

Above, Bob requests an access in the capacity of "*Alice* as *IG.user*." Note that *access* is not an entity. This is a delegation from Bob to a dummy entity representing the request access. RT^D assigns a unique dummy entity to each request.

We conjecture that other extended forms of RT can be shown to be similarly applicable to additional use cases within the community cyber security information sharing domain.

V. Administrative Limitations of RT

We have incrementally developed a series of RT-based formal models for community cyber security information sharing. In this process we have discovered some representative limitations of the RT approach as described below.

Entity-Owned Only Membership: We found it necessary to augment RT with the notion of an open role $\mathit{OG}.\underline{\mathit{volunteer}}$ to enable a user to join a role at user's volition. We believe this will be a significant feature in future cyber systems. Forcibly enrolling all eligible members into a volunteer group is not appropriate. This is better managed as a user-driven process. While the enhancement is small with essentially no impact on RT's theoretical analysis and security properties, its impact on administrative policies is very significant.

Reverse Credential Chains: RT uses a method to discover the credentials of a member which only propagate in bottom-up fashion. However, a highly dynamic distributed system requires two way information flow. In our construction the role name *cgrep* was introduced solely to enable organizations such as *CPS*, *SAWS* and *SAPD* to appoint their representatives to the core group *CG*. Moreover this is open only to organizations who are in *SAT.member*. Suppose the entity *SAT* decides to drop *SAWS* from *SAT.member*. Then the role *SAWS.cgrep* becomes useless and may as well be dropped from *SAWS*. There is no mechanism in RT to facilitate such communication.

Unable to Support Administration from an External Entity: In RT, an entity is the only authority to manage the membership of all the roles belongs to that entity. However, in some distributed authorization systems, a role membership might need to be controlled from a parent or external entity. For instance, in our model, members of *IG.user* need to be selected, in part, by the entity *CG*. Further we may wish enable *CG* to create and destroy different *IG*'s. The entity-controls-all approach of RT does not facilitate this process.

Continuous Enforcement of Membership Conditions: RT appears to have difficulty imposing a restriction only at role assignment time but not thereafter. Policy statements (22) and (23) ensure that a member of *IG.user* is authorized by *IG* and at the same time they impose another condition that he needs to be a member of either *CG* or *OG*. Note that the ARBAC97 model [8] enforces its concept of prerequisite conditions only at role-assignment time. A more general administrative model may need to accommodate both approaches.

VI. Conclusion and Discussion

We have developed a series of RT-based formal administrative models for a set of recently proposed informal requirements for community cyber security information sharing. This has required a small but significant enhancement of the entity-oriented RT approach. Specifically, the concept of an open role where users can self-enroll is required to support an essential feature of this application domain. With this enhancement RT is substantially adequate for the task. Some of the extended forms of RT are also clearly useful. Nonetheless, the RT approach to administration seems to have some fundamental limits. Moreover RT is not operationalized to the extent that a practical implementation would require, e.g., by incorporating explicit revoke statements. It remains a research challenge to incorporate the desirable and effective features of RT in a bigger framework that addresses its shortcomings. Note that algorithms for credential chain discovery in RT have been implemented [6]. In future work the formal models developed in this paper for community cyber security can be further refined and demonstrated in practical implementations, perhaps embodied as software-as-a-service.

Acknowledgment

The authors are partially supported by grants from AFOSR MURI and the State of Texas Emerging Technology Fund.

References

[1] R. Bhatti, E. Bertino, and A. Ghafoor. An integrated approach to federated identity and privilege management in open systems. *Commun. ACM*, 50:81–87, February 2007.

[2] R. Krishnan, J. Niu, R. S. Sandhu, and W. H. Winsborough. Group-centric secure information sharing models for isolated groups. *ACM Transactions on Information and System Security*, to appear.

[3] R. Krishnan, R. S. Sandhu, J. Niu, and W. H. Winsborough. Foundations for group-centric secure information sharing models. In *ACM SACMAT*, pages 115–124, 2009.

[4] R. Krishnan, R. S. Sandhu, J. Niu, and W. H. Winsborough. Towards a framework for group-centric secure collaboration. In *IEEE CollaborateCom*, pages 1–10, 2009.

[5] N. Li, J. C. Mitchell, and W. H. Winsborough. Design of a role-based trust management framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 114–130. IEEE Computer Society Press, May 2002.

[6] N. Li, W. H. Winsborough, and J. C. Mitchell. Distributed credential chain discovery in trust management: extended abstract. In *Proceedings of the 8th ACM conference on Computer and Communications Security*, CCS '01, pages 156–165, New York, NY, USA, 2001. ACM.

[7] N. Li, W. H. Winsborough, and J. C. Mitchell. Distributed credential chain discovery in trust management. *Journal of Computer Security*, 11(1):35–86, Feb. 2003.

[8] R. Sandhu, V. Bhamidipati, and Q. Munawer. The ARBAC97 model for role-based administration of roles. *ACM Trans. Inf. Syst. Secur.*, 2:105–135, February 1999.

[9] R. Sandhu, R. Krishnan, and G. White. Towards secure information sharing models for community cyber security. In *Proc. 6th IEEE International Conference on Collaborative Computing*, 2010.

[10] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29:38–47, 1996.

[11] D. Shands, R. Yee, J. Jacobs, and E. Sebes. Secure virtual enclaves: supporting coalition use of distributed application technologies. In *DARPA Information Survivability Conference and Exposition*, 2000.