A Group-Centric Model for Collaboration with Expedient Insiders in Multilevel Systems

Khalid Zaman Bijon*, Ravi Sandhu* and Ram Krishnan†
*Institute for Cyber Security & Department of Computer Science
†Institute for Cyber Security & Department of Electrical and Computer Engineering
University of Texas at San Antonio

***Abstract*—An authorization model for group-centric organizational collaboration has been recently proposed wherein multiple organizations may collaborate via groups [3]. Each group is independent of all others and adheres to the formal semantics of Group-Centric Secure Information Sharing models (g-SIS) [2], [4]. Motivated by [3], in this paper, we develop a model for group-centric collaboration in which an organization forms groups to collaborate with outside consultants on specific projects. A core principle is that such outsiders cannot fit in the existing organizational access control structure as they are not "true insiders" but rather "expedient insiders." In our proposed model, each group duplicates the organizational access control structure in an identical but separate copy—initially without any assignment of users or objects. The group is then populated and maintained by bringing selected true insiders, expedient insiders, and objects together to enable collaboration. The formal model consists of administrative and operational parts covering the complete life-cycle. While the general concepts are applicable regardless of the specific models used for the organizational access control structure, to be concrete we consider the specific case of multilevel systems that enforce lattice-based access control [7].**

Keywords- Group Centric Collaboration; Information Sharing; Lattice Based Access Control;

I. Introduction

Organizational collaboration with outsiders, such as domain specialists and other experts, is commonplace in practice today. During the period of collaboration, such outsiders are assigned necessary privileges to access sensitive information objects of the organization to meaningfully contribute to the project. Currently, there is no well-accepted model for this type of information sharing or privilege management. Consequently such sharing is often ad hoc. With the anticipated increase in such collaborations the need for rigorous models in this arena will also increase. In this paper we develop a formal model for this purpose. A core principle of the model is that outsiders should never be granted authorization credentials, such as security clearances or roles, identical to those granted to the organization's employees. As such these outside collaborators are not "true insiders" but rather "expedient insiders" who should receive much more limited privileges.

Most organizations maintain an access control structure that best fits their requirements to control information flow among true insiders. One such organizational structure is the lattice-based model (also commonly known as multilevel security or mandatory access control). In a lattice-based model, persons are cleared and objects are classified in different security classes, such as Top-Secret (TS), Secret (S), Confidential (C) and Unclassified (U). Here information flows in a total or linear ordering of those security classes where TS $>$ S $>$ C $>$ U (more generally security classes are partially ordered). For example, a member with clearance to a point of lattice, e.g. S, may only get access to an object that is classified as same or lower level, i.e. S, C or U. This information sharing is based on the long-term relationship between the organization and its members as well as members' responsibility, trustworthiness and accountability. Thus, this inner organizational structure does not work properly for those collaborative outsiders as they are transient rather than persistent. They do not have same accountability as the regular members have, nor do they posses the same level of trustworthiness. In other word, these outsiders do not fit to the regular organizational structure because they are not true insiders but expedient insiders. Usually, organizations need to share only selected information for the purpose of a collaboration, therefore, allowing those expedient insiders in regular organizational framework might unnecessarily expose sensitive information.

In this paper, we focus on collaboration between a single organization and outside independent consultants. We analyze several different processes to establish this collaboration and demonstrate that the Group-Centric concept is appropriate for this purpose. Group-Centric sharing was introduced in [4] where the primary focus was on the temporal semantics of group operations. In a group, users may join, leave or re-join and objects may be added, removed and re-added. User's access to an object depends on the temporal ordering of these events. However, [4] deliberately does not specify administrative aspects such as who authorizes these operation, who manages the membership, etc.

In [3], a model for group centric collaboration in inter-organizational scenarios was introduced consisting of administrative and operational components. Three distinct collaborative scenarios are discussed, viz., bilateral organizational collaboration, unilateral organizational collaboration and qualified open collaboration. In bilateral collaboration, two or more organizations collaborate with each other for some common purposes such as developing intellectual property. In unilateral collaboration, there is a single central organization which

collaborates with external parties on its own terms. The main contribution of [3] was to provide a complete model for bilateral collaboration. Their proposed group model does not incorporate any hierarchical access control structure.

The main contribution of our paper is to provide a group-centric model for unilateral organizational collaboration, particularly for collaboration with external individuals. Each collaboration group incorporates the organizational access control structure. For simplicity and concreteness, we assume that this structure is a security lattice with one-directional information flow [7]. More specifically we assume that the usual simple-security property applies for read operations and that the strict form of the star-property applies for write operations, wherein write is only permitted at the security level of the subject attempting the write. Our model can be easily extended to other access control structures such as role-based access control, attribute based access control, etc. In our model, a established group uses the identical security lattice as the underlying organization but with different users and objects.

The remainder of this paper is as follows. Section II discusses different processes to establish collaboration groups. A model for collaboration with expedient insiders is formally specified in section III and section IV gives our conclusions.

II. Different Processes for Organizational Collaboration with Outsiders

In this section we first outline a collaboration scenario with outsiders and then analyze three different processes to integrate this collaboration with an organization.

Consider a scenario where an organization Org with sensitive information, such as a defense contractor, needs to consult with outside specialists for a specific reason R. E is a set of consultants providing service for R and the organization may want to collaborate with some of them. Let Org select a set of true insiders T and a set of consultants (expedient insiders) C for this collaboration. Members in T and C might change over time based on fluctuating purpose and state of the collaboration. In such a collaboration Org is the ultimate stake holder and consultants are only compensated for their service. Organization also needs to select objects and grant necessary access to the users in T and C. This authorization process should be efficient and able to protect unauthorized disclosure of sensitive information. Therefore, the main challenge is to fit the consultants in a particular place in the organizational framework where they only get access to the necessary documents and their presence does not unnecessarily affect other sections of the organization. Below we discuss three different possible processes towards this goal.

A. Assign Outsiders at a Point in Organizational Structure

One possibility is to place an expedient insider at a suitable point in the already existing organizational structure. In a lattice structure this amounts to temporarily assigning the expedient insider a clearance equal to one of the labels in the lattice. There are a number of drawbacks to this approach. Firstly, only a subset of information accessible by true insiders at a given clearance might be necessary for a collaboration. Therefore, the expedient insider may obtain access to other sensitive information which is not relevant to the purpose of the collaboration. Again, presence of an expedient insider on equal footing as true insiders may create ambiguity and concern for other true insiders unnecessarily impacting the regular information flow of the organization.

A possible extension of this approach is to modify the lattice to incorporate additional security labels which are reserved for collaborating true insiders and expedient insiders. We believe this runs counter to the culture of access control where security architects would strongly prefer a stable structure. Nevertheless this approach may be of theoretical interest.

B. Individual Sharing Collaboration

An organization may collaborate with outsiders individually and separately without assigning them a clearance in the lattice. During collaboration phase, the organization may share some objects with each individual on a need-to-know basis with risk-benefit judgement. This is perhaps the simplest approach for such collaborations. The main problem here is scalability. This procedure may be effective for micro-collaboration where a single collaborator is enough. However, it fails to provide service for large scale collaboration because this collaboration might need a number of consultants. In large scale collaboration, each collaborator might get same sets or some common subsets of objects. Maintaining consistency amongst these may be difficult, so some consultants are inadvertently denied essential information while others get too much. As new or replacement consultants are brought on board mistakes may be made in providing necessary information to them. Organizations often split a big project into several small parts and each part may need different consultants. If the collaboration is organized as individualized sharing it may be difficult to merge scattered results to generate a final product.

C. Group-Centric Collaboration

The Group-Centric approach for inter-organizational collaboration was proposed in [3] where an authorization model was introduced for collaboration groups formed by organizations with common interests. Our analysis indicates that this Group-Centric approach is also suitable for collaboration with outsiders for a number of reasons. An organization can establish a group for the purpose of collaboration and bring necessary objects, selected true insiders and expedient insiders together in the group. Note that, within an organization, true insiders already will have access to the selected objects provided to the collaboration group (within the limits of their security labels). However, assigning true insiders to a collaboration group further specializes their responsibility for a specific collaboration and makes explicit what is being shared with

expedient insiders. Moreover, in the collaboration members can create new objects or update previously added objects from the organization. These objects will not, and should not, be accessible by the true insiders of the organization until they are explicitly imported into the organization. Therefore, adding true insiders in a group authorizes them instant access to such information that is necessary to meaningful collaboration.

Administrative management of such groups, including their establishment and disbanding, can be conveniently carried out and monitored by an organization. Users share objects within a group and can edit and update them in a consistent manner that solves the scalability problem of individual sharing. A group administrator can add objects to the group from any security label whenever it is necessary, while keeping the objects protected within the group to users with proper clearance. Thus, the organization can be more selective in information sharing with expedient insiders so this process is more secure than placing those expedient insiders within the organizational structure. Groups also confine member's write permission to the group thereby restricting malicious subjects, i.e. trojan horses, to leak confidential information. Finally, presence of a collaboration group does not disrupt the organization's original structure because the group does not share the organization's internal information flow. Access via the group is limited strictly to those objects made available to the group. Objects modified within the group or created within the group are not visible to true insiders outside the group until they are explicitly imported or merged back into the organization.

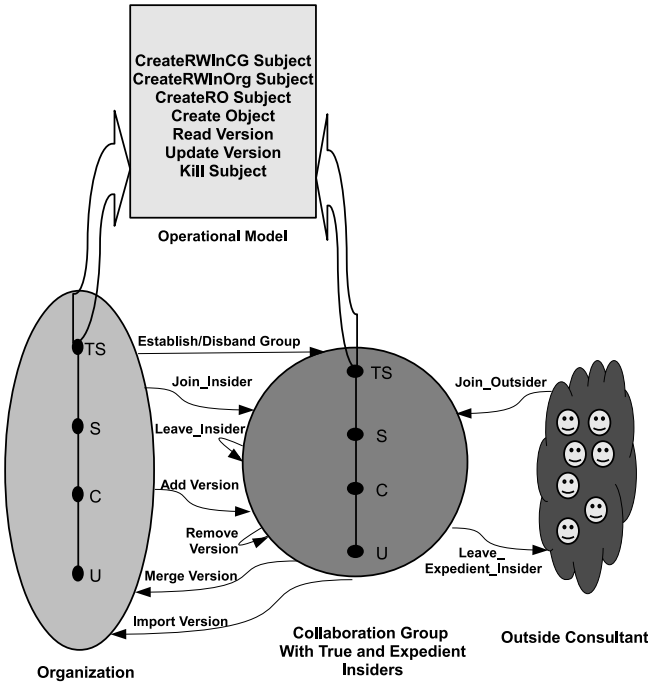

Figure 1. *Operational Semantics of Collaboration Group with Outsiders. The group and the organization have the same security lattice, e.g., Top Secret (TS), Secret (S), Classified (C), Unclassified (U). Named arrows show the Administrative operations. User operations are shown in the rectangle at the top.*

III. Formal Specification of Organizational Collaboration with Expedient Insiders

In this section a group-centric policy model is formally specified for this specific organizational collaboration with expedient insiders. In this model, relevant attributes of the entities are exploited to authorize each operation. For example, before an operation (such as writing an object) is permitted all the involved entities' (e.g. user, object, etc) attributes (user id, object type, etc) are evaluated in order to approve it. For this purpose we follow the well-known UCON model [5] and specifically its pre-authorization component. UCON provides mutable attributes which are updated dynamically as appropriate, as each operation is authorized. For example, if the operation is to merge two versions of an object, the "object version" attributes are updated accordingly. Details of the attributes definition are provided in III-B. The remainder of this section contains the model overview, the attribute definitions, the administrative model, and the operational model. For convenience, we use the term "group" for collaboration group, "Org" for the organization, "true insider" for a group member from Org, "group administrator" or "group admin" for administrative members of the group and "expedient insider" for outside consultants.

A. Overview

An organization may establish any number of distinct collaboration groups, all of which use the identical lattice structure as the organization. True insiders always carry the same security clearance as they have in the organization. Each expedient insider is assigned a suitable security clearance when she is assigned to a group. The same clearance will apply if she is added to additional groups. The group administrator has complete control on which users (true insiders or expedient insiders) and objects are members of the group.

In this distributed collaborative setting we assume that objects are versioned. In a general version model each write creates a new version of an object. A group administrator can bring specific versions of selected objects from the organization into the group. These versions retain their previous security classification in the group. For simplicity we assume that all version of an object are at the same security classification. Expedient insiders are not permitted to add any objects from outside. They can only contribute via newly created and modified objects in the group. New objects, and new versions of existing objects, may be created in the group during collaboration. The group administrator may import specific version of those objects to the organization.

TABLE I
ATTRIBUTE SPECIFICATION

Global Sets and Symbols: SL: Finite lattice of security levels with dominance ordering $\succeq$ CG: Finite set of existing groups U: Finite set of existing users O: Finite set of existing objects S: Finite set of existing subjects UNIV_V: The universal set of versions (an infinite set) Org: The organization (a constant symbol)
User Attributes: Att(U)={clearance, ucg, orgadmin, cgadmin, utype} clearance: U $\rightarrow$ SL ucg: U $\rightarrow 2^{CG}$ orgadmin: U $\rightarrow$ {True, False} cgadmin: U $\rightarrow 2^{CG}$ utype: U $\rightarrow$ {Insider, Expedient_Insider, null}
Objects Attributes: Att(O)={classification, origin, versions} classification: O $\rightarrow$ SL origin: O $\rightarrow$ CG $\cup$ {Org} versions: O $\rightarrow 2^{UNIV\_V}$
Subject Attributes: Att(S)={clearance, owner, belongsTo, type} clearance: S $\rightarrow$ SL owner: S $\rightarrow$ U belongsTo: S $\rightarrow$ CG $\cup$ {Org} type: S $\rightarrow$ {RW,RO}
Object Version Attribute: Att(O, V)={vMember, classification} vMember: O $\times$ UNIV_V $\rightarrow 2^{CG \cup \{Org\}}$ classification: O $\times$ UNIV_V $\rightarrow$ SL /* These are partial function defined only for the versions that exist for each object*/

Figure 1 informally illustrates the group model for such collaborations. The group uses the same lattice structure as the organization. Administrative operations are indicated by named arrows. The arrow end indicates the principal target of the operation. The operations listed within the rectangle at the top comprise the user operations, for both true and expedient insiders. Depending on the security labels, these operations are authorized to different users for different sets of objects.

B. Attributes

The sets and attributes used in this collaboration system are given in Table I. The organization is represented by the constant symbol Org. SL is the lattice of security labels ordered with the dominance relation $\succeq$. CG, U, O and S respectively represent the set of current collaboration groups, users, objects and subjects. UNIV_V is the universe of all version names. At any moment each existing object will have some finite subset of UNIV_V as its existing versions. This subset will be different for different objects at a given moment.

User Attributes: A user is a human being in the system. Users are of type Insider or Expedient_Insider as specified by the user attribute utype. A user's clearance is represented by the clearance attribute that contains a security label from SL. Attribute ucg contains the set of groups where the user is a member. Attribute cgadmin maintains the set of groups where the user is an administrator. Attribute orgadmin specifies whether or not the user is an organization administrator.

Object Attributes: Attribute origin represents the group or Org where the object was created. An object's security classification is specified by the classification attribute. The versions attribute maintains the set of versions of an object.

Subject Attributes: Attribute owner represents the user who created the subject. The group or Org where the subject is created is specified by attribute belongsTo. A subject is defined as either read-only or read-write by the type attribute.

Object Version Attributes: Each object can have a number of versions and each version may be shared by one or more groups. Thus, for each object and version pair, the attribute vMember lists entities (i.e., groups and/or Org) that share it. Attribute classification indicates the security classification of the version. All versions of an object have the same classification value, so strictly speaking this attribute is redundant.

C. Administrative model

Different semantics of four core group operations (user Join and Leave and object Add and Remove) have been proposed in [4]. A user may join a group in two different ways: Strict

TABLE II
ADMINISTRATIVE MODEL

Operation	Precondition	Updates
Establish(u, cg) /*Admin user u establishes new collaboration group cg*/	$u \in U \wedge cg \notin CG \wedge$ orgadmin(u)=True	$cgadmin'(u) = cgadmin(u) \cup \{cg\}$ $CG' = CG \cup \{cg\}$
Join_Insider(u1,u2,cg) /*Admin u1 grants cg membership to a true insider u2*/	$u1 \in U \wedge u2 \in U \wedge cg \in CG \wedge$ $cg \in cgadmin(u1) \wedge$ $utype(u2) = Insider \wedge cg \notin ucg(u2)$	$ucg'(u2) = ucg(u2) \cup \{cg\}$
Leave_Insider(u1,u2,cg) /*Admin u1 revokes cg membership from a true insider u2*/	$u1 \in U \wedge u2 \in U \wedge cg \in CG \wedge$ $cg \in cgadmin(u1) \wedge cg \in ucg(u2)$ $\wedge$ utype(u2) = Insider	$ucg'(u2) = ucg(u2)$ - {cg} ***forall*** s ∈ S ***if*** owner(s) = u2 ∧ belongsTo(s) = cg ***then*** S′ = S - {s} /*Kill subjects in cg owned by u2*/
Join_Outsider(u1,u2,cg,seclab) /*Admin u1 grants cg membership to an expedient insider u2*/	$u1 \in U \wedge u2 \in U \wedge cg \in CG \wedge$ $cg \in cgadmin(u1) \wedge cg \notin ucg(u2)$ $\wedge$ utype(u2)$\neq$ Insider	$utype'(u2) = Expedient\_Insider$ ***if*** $ucg(u2) = \emptyset$ ***then*** $clearance'(u2) = seclab$ $ucg'(u2) = ucg(u2) \cup \{cg\}$
Leave_Expedient_Insider(u1,u2,cg) /*Admin u1 revokes cg membership from an expedient insider u2*/	$u1 \in U \wedge u2 \in U \wedge cg \in CG \wedge$ $cg \in cgadmin(u1) \wedge cg \in ucg(u2)$ $\wedge$ utype(u2) = Expedient_Insider	$ucg'(u2) = ucg(u2)$ - {cg} ***forall*** s ∈ S ***if*** owner(s) = u2 ∧ belongsTo(s) = cg ***then*** S′ = S - {s} /*Kill subjects in cg owned by u2*/
Add(u,o,v,cg) /*Admin u adds version v of object o from Org to cg*/	$u \in U \wedge cg \in CG \wedge o \in O \wedge$ $v \in versions(o) \wedge cg \in cgadmin(u) \wedge$ $cg \notin vMember(o,v)$	$vMember'(o,v) = vMember(o,v) \cup \{cg\}$
Remove(u,o,v,cg) /*Admin u removes version v of object o from cg*/	$u \in U \wedge cg \in CG \wedge o \in O \wedge$ $v \in versions(o) \wedge cg \in cgadmin(u) \wedge$ $cg \in vMember(o,v)$	$vMember'(o,v) = vMember(o,v)$ - {cg}
Import(u,o1,v1,o2,cg) /*Admin u imports version v1 of object o1 to new version v2 of object o2 in Org*/	$u \in U \wedge cg \in CG \wedge v1 \in versions(o)$ $\wedge$ o1,o2 ∈ O ∧ origin(o2) = Org ∧ $cg \in cgadmin(u) \wedge origin(o1) = cg \wedge$ classification(o1) = classification(o2)	$versions'(o2) = versions(o2) \cup \{v2\}$ /*v2 is newly created version of o2*/ $vMember'(o2,v2) = \{Org\}$ classification(o2,v2) = classification(o2)
Merge(u,o,v,cg) /*Admin u merges version v of object o from cg to Org*/	$u \in U \wedge cg \in CG \wedge o \in O \wedge$ $v \in versions(o) \wedge cg \in cgadmin(u) \wedge$ $cg \in vMember(o,v) \wedge$ $origin(o) = Org \wedge v \in versions(o)$	$vMember'(o,v) = vMember(o,v) \cup Org$
Disband(u, cg) /*Admin u disbands a collaboration group cg*/	$u \in U \wedge cg \in CG \wedge cg \in cgadmin(u)$	***forall*** u1 ∈ U ***if*** cg ∈ ucg(u1) ***then*** $ucg'(u1) = ucg(u1)$ - {cg} ***if*** cg ∈ cgadmin(u1) ***then*** $cgadmin'(u1) = cgadmin(u1)$ - {cg} ***forall*** o ∈ O ***if*** origin(o) = cg ***then*** O′ = O - {o} ***forall*** o ∈ O ***forall*** v ∈ versions(o) ***if*** cg ∈ vMember(o,v) ***then*** $vMember(o,v)' = vMember(o,v)$ - {cg} CG′ = CG - {cg} $S' = S - \bigcup_{\forall s \in S.belongsTo(s)=cg} S$

Join (SJ) or Liberal Join (LJ). Users with SJ may only access objects that are added after their join time. With LJ, users may also access objects added prior to join time. Objects could be added to a group in two ways: Strict Add (SA) and Liberal Add (LA). In LA, all the present and future members of the group can access the object. In SA, access is limited to present members of the group at the time of SA. Similar to Join, users may also leave with a Strict Leave (SL) or Liberal

Leave (LL). In SL, users lose complete access to all objects in the group where in LL, the users retain access to objects authorized to them during membership period. Further, objects could be removed in strict and liberal fashion. In Strict Remove (SR), group users will not able to access the removed object after remove time where Liberal Remove (LR) allows those users to retain access permission. Any combination of these semantics may have different impacts on access relationship between subjects and objects. For simplicity, we confine the authorization semantics to Liberal Join, Strict Leave for users and Liberal Add, Strict Remove for objects. This is the typical semantics used for groups in traditional access control systems. Note that any variation of these semantics will only affect the authorizations of Read and Update operations leaving the rest of the specification essentially intact.

True insiders and objects retain their organizational security classifications in a group. However, the administrative model requires assignment of appropriate security clearance to every newly joined expedient insider. Note that the clearance of each user remains the same in all groups. Similarly, each object and its different versions always have same classification across different groups and in the Org.

Table II formally specifies a set of administrative operations for this collaboration group. The first column specifies the operation that is to be performed. The second column specifies the conditions that need to be satisfied to authorize the operation. Attributes and sets that will be updated after an authorized operation are listed in the third column, with the $'$ symbol indicating the value after the update. These administrative operations are discussed below.

- **Establish** collaboration group: An organization forms a collaboration group and an administrative user from the organization is appointed as group administrator. First column of table II specifies the operation name and parameters. Here u is the selected administrative user and cg is the new group name. The second column specifies that cg should be a new group name that does not currently exist in CG, and that user u is a valid administrative user of Org. Finally, in the third column, u becomes the cg administrator and cg is added to the existing group set CG.
- **Join_Insider** to group: Group administrator u1 can grant membership to a true insider u2 in a group cg. An authorization process checks whether u1 and u2 are group administrator and true insider respectively for this operation. If so cg is added to the ucg attribute of u2.
- **Leave_Insider** from group: Revokes group membership of a true insider. Further, it kills subjects that were created by that insider in the group cg.
- **Join_Outsider** to group: Group administrator u1 can enroll an outsider u2 as an expedient insider who does not have current group membership. However, he might hold membership to other collaboration groups of the organization. In that case, u2 retains the same security clearance for this group. Otherwise, group administrator u1 assigns an appropriate security clearance.
- **Leave_Expedient_Insider** from group: Revokes group membership of an expedient insider. Further, it kills subjects that were created by that insider in the group cg. Note that if this results in ucg(u2) becoming empty then on some future join, should it occur, the security clearance of this user will be set to a new value.
- **Add** an object version to group: Administrator of a group adds a version v of an object o from the organization to the group. The vMember attribute of version v of object o is updated accordingly.
- **Remove** a version from group: Using this operation group administrator removes an object version from a group cg. Accordingly cg is removed from the list vMember of that version.
- **Import** a version from group to Org: A version v1 of an object o1 can be copied to Org from a group cg by the group administrator. This operation can only be performed to an object o1 that is natively created in cg, whereby origin(o1)=cg. The group administrator copies the object version o1, v1 to a new version v2 of o2 where origin(o2)=Org. Note that, classification of o1 and o2 should be equal for a successful import.
- **Merge** a version to Org: The semantics of this operation is to merge back a new object version v of o from cg to Org where origin(o)=Org. If this operation is successful, version v of o includes Org in its membership set vMember.
- **Disband** group: The semantics of this operation is to delete the group cg. Prior to Disband the group administrator will Merge and Import necessary objects from the group to the organization. After Disband, the corresponding attributes of every true insider, expedient insider and object version are updated accordingly. Every subject executing in the group needs to be killed and and every object with origin cg needs to be deleted. The disbanded group is removed from the cgadmin attribute of all users. Finally, cg is removed from CG.

D. Operational model

Table III specifies the set of user operations involved in operational model. Using these operations a user can create subjects and exercise privileges in a group or Org. A subject is a program or process that runs in the system on behalf of a user. A subject inherits the same or lower security clearance from the user who created it. A user may create multiple subjects, however, a subject is owned by only one user. We assume, a particular insider (both true and expedient) can be a member of more than one group. For the purpose of aggregation, a user can create a read-only subject which can read an object version from any group to which the user belongs and/or the Org (for true insiders). A read-only subject is unable to write. In order to write a user must create a read-write subject which is confined to write either only in a single

TABLE III
OPERATIONAL MODEL

Operation	Precondition	Updates
CreateRWInCG(u,s,cg,seclab) /*User u creates read-write subject s in a group cg*/	$u \in U \wedge s \notin S \wedge cg \in ucg(u) \wedge$ $seclab \preceq clearance(u)$	$owner'(s) = u$ $clearance'(s) = seclab$ $belongsTo'(s) = cg$ $type'(s) = RW$ $S' = S \cup \{s\}$
CreateRWInOrg(u,s,seclab) /*Only true insider creates read-write subject in Org*/	$u \in U \wedge s \notin S \wedge utype(u) = Insider \wedge$ $seclab \preceq clearance(u)$	$owner'(s) = u$ $clearance'(s) = seclab$ $belongsTo'(s) = Org$ $type'(s) = RW$ $S' = S \cup \{s\}$
CreateRO(u,s,seclab) /*User u creates read-only subject s*/	$u \in U \wedge s \notin S \wedge seclab \preceq clearance(u)$	$owner'(s) = u$ $clearance'(s) = seclab$ $type'(s) = RO$ $S' = S \cup \{s\}$
Read(s,o,v) /*Subject s reads the version v of object o*/	$s \in S \wedge o \in O \wedge v \in versions(o) \wedge$ $clearance(s) \succeq classification(o,v) \wedge$ $(type(s) = RO \wedge$ $((ucg(owner(s)) \in vMember(o,v)) \vee$ $(utype(owner(s)) = Insider \wedge$ $\{Org\} \in vMember(o,v)))) \vee$ $(type(s) = RW \wedge$ $(belongsTo(s) \in vMember(o,v))))$	None
Update(s,o,v) /*Subject s updates the version v of object o. This function returns updated version v1*/	$s \in S \wedge o \in O \wedge v \in versions(o) \wedge$ $clearance(s) = classification(o,v) \wedge$ $(type(s) = RW \wedge$ $belongsTo(s) \in vMember(o,v))$	$versions'(o) = versions(o) \cup \{v1\}$ /*v1 is newly created version id*/ $vMember'(o,v1) = vMember(o,v1) \cup \{cg\}$ $classification'(o,v1) = classification(o,v)$
Create(s,o) /*Subject s creates version v of object o. This function returns newly created version v*/	$s \in S \wedge o \notin O \wedge type(s)=RW$	$O' = O \cup \{o\}$ $versions'(o) = \{v\}$ /*v is newly created root version id*/ $vMember'(o,v) = \{belongsTo(s)\}$ $origin'(o) = belongsTo(s)$ $classificaton'(o) = clearance(s)$ $classificaton'(o,v) = clearance(s)$
Kill(u,s) /*User u kills subject s*/	$u \in U \wedge s \in S \wedge$ $owner(s) = u \vee belongsTo(s) \in cgadmin(u)$	$owner'(s) = Null$ $type'(s) = Null$ $clearance'(s) = Null$ $belongsTo'(s) = Null$ $S' = S - \{s\}$

group or only in the Org, depending on where it was created. This is a critical aspect of our model. The scope of a subject's write operation is further restricted by the strict star-property within the single group or Org.

- **CreateRWInCG** or **CreateRWInOrg** subjects: A user can create read-write subject in Org or in a cg where she is a member. In order to create a read-write subject in Org the user needs to be a true insider. The user also determines the security clearance of the subject. However, the user can only specify the subject clearance as less than or equal to user's own security clearance. A subject belongsTo to the group or Org in which the user creates it. The type attribute of the subject is RW.
- **CreateRO** subjects: A user can also create a read-only subject. Similar to a read-write subject, the subject's security clearance must be dominated by the user's clearance. The type attribute of this subject is RO.
- **Read** version of an object: Read is one of the most critical operations of the system. In order to aggregate information, a RO subject can read any object version from every group and/or Org in which the subject's owner is a member, while a RW subject is restricted to read only in the group or Org in which it is created. By the simple security policy, in order to authorize a read the clearance of a subject must be higher or equal to the classification of the target version.

- **Update** version of an object: Each update creates a new version of an object. Only a RW subject can perform an Update. Authorization process of this operation depends on the clearance of the subject compared to the classification of the target object. Our model enforces strict star-property in which object's classification should be equal to the subject's clearance for a successful Update.
- **Create** object: A RW subject can create a new object in the group or Org where it was created. A newly created object inherits the security classification from respective subject's clearance.
- **Kill** subject: Kill operation can be performed either by the owner of the subject or the group administrator where the subject resides. This operation removes the subject from the set of subjects S.

E. Other Related Works

Several authors have addressed security issues in information sharing among organizations. A number of security issues for the Dynamic Coalition Problem (DCP) are discussed in [6] for coalitions formed internationally in response to a crisis. Their model defines involved organization types and roles in such a coalition and proposes a candidate security approach that meets the goal of DCP. Warner et al [8] proposed a dynamic coalition-based access control model that facilitates users from one coalition entity to automatically access another entity. In this model, a service registry contains coalition level access policies for each entity and for accessing other entity's resources one must obtain necessary ticket from the registry. The registry also has its own access control policy to verify the credentials of a requestor in order to approve the ticket. O2O [1] combined two concepts: virtual private organization and role single-sign on for secure information sharing among organizations. However, none of these models addressed the issues for organizational collaboration with expedient insiders as defined in this paper.

F. Model Enhancement

In this section we discuss possible enhancements of the capability of different operations that we have proposed in our group-centric collaboration model.

Join_Insider could modify clearance: In our proposed administrative model, a true insider retains his organizational clearance unmodified in every group he joins. However, in practice, a true insider might get a different security clearance in a group. For example, an organization may choose a junior employee as a leader of a collaboration group. That employee might have lower security clearance in organization while he needs higher clearance in the group. Conversely a highly cleared true insider may be given a lower clearance in a group depending on the role of that use in the collaboration.

Join_Outsider could allow different clearances for the same expedient insider in different collaboration groups. This accords with the principles of least privilege and need-to-know.

Add could sanitize information of an object: An object might contain very sensitive information that makes it unsuitable to share with expedient insiders in a collaboration group. However, some sanitization or redaction could make it appropriate to share. For instance, an organization may not be willing to share every employee's salary, while it might share the average salary.

IV. Conclusion

Our goal in this paper is to define a suitable formal model for collaboration with expedient insiders. To this end, we adapted the group-centric model due to its potential to exploit many essential features in this collaboration. For instance, a group enables selective information sharing with expedient insiders on need-to-know basis. A collaboration group can conveniently determine a place for expedient insiders in the organization without disrupting the main organizational structure. In this paper, we have also proposed an authorization model which consists of separate administrative and operational components.

In the future, we are interested to develop a model for group-centric multi-organizational collaboration where in a group every organization's structure might merge in a certain point and members from those organizations, as well as, expedient insiders get proper privileges. Participating organizations may have different structures, therefore, it is a challenge to find a proper supporting framework for such group that is flexible enough to manage members and objects from different organizations.

Acknowledgment

This work is partially supported by grants from AFOSR MURI, the State of Texas ETF and NSF.

References

[1] F. Cuppens, N. Cuppens-Boulahia, and C. Coma. O2O: Virtual private organizations to manage security policy interoperability. In *Information Systems Security*, volume 4332, pages 101–115. Springer, 2006.

[2] R. Krishnan, J. Niu, R. Sandhu, and W. H. Winsborough. Group-centric secure information-sharing models for isolated groups. *TISSEC*, 14, 2011.

[3] R. Krishnan, R. Sandhu, J. Niu, and W. Winsborough. Towards a framework for group-centric secure collaboration. In *CollaborateCom*, 2009.

[4] R. Krishnan, R. Sandhu, J. Niu, and W. H. Winsborough. Foundations for group-centric secure information sharing models. In *SACMAT*, 2009.

[5] J. Park and R. Sandhu. The UCON$_{ABC}$ usage control model. *ACM Trans. on Information and Systems Security*, 7:128–174, 2004.

[6] C. E. Phillips, T. C. Ting, and S. A. Demurjian. Information sharing and security in dynamic coalitions. In *SACMAT*, 2002.

[7] R. S. Sandhu. Lattice-based access control models. *IEEE Computer*, 26:9–19, 1993.

[8] J. Warner, V. Atluri, and R. Mukkamala. A credential-based approach for facilitating automatic resource sharing among ad-hoc dynamic coalitions. In *DBSec*, 2005.