Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 8 (20): 3561-3571, 2008
ISSN 1812-5654

Modeling and Combining Access Control Policies Using Constrained Policy Graph (CPG)

Z. Derakhshandeh, B.T. Ladani and N. Nematbakhsh
Department of Computer Engineering, University of Isfahan, Isfahan, Iran

Abstract: In this study, a formal model for specification of Access Control Policies (ACP) is represented. This model is capable of expressing several ACPs and combining them in a unified framework. We call present model Constrained Policy Graph (CPG), which is an extension of Take-Grant (TG) protection model. Although TG can be used to specify the ACPs but it represents the policies without any constraint (e.g., time, location, or any other restriction parameters). Furthermore, it hasn't ever been used for combining the policies and nested expression of them. In present proposed model, not only the policies can be constrained according to system requirements but also, it can be used for combining ACPs as well as their nested specification. Furthermore, ACPs can be verified conflicts or contradictions using this model. One of the main applications of the proposed model is specifying and combining the ACPs of web services and verifying their composed policies in web service composition.

Key words: Policy combination, verification, conflict, constraints

INTRODUCTION

Protecting system resources against the access of unauthorized people is the main goal of access control systems. Many studies have been done to specify the Access Control Policies (ACPs) in different ways. Generally, it is required to unify and combine several policies to specify a system policy according to its subsystems policies. In other words, a single policy normally cannot provide the required protective level of a real system. So, there is a need to have a method for combination of policies. In spite of noticeable improvements in the access control field, most of the methods are based on specifying and applying single policies, rather than unified integrated policies for modeling the systems policies. A few number of them support ACP combination but they do not provide all requirements for covering complete specification and combination of ACPs. In addition to the combination requirement, presenting a method for verification of conflict or contradiction in the combined policies is essential.

This study is proposed the Constrained Policy Graph (CPG) model, which enjoys Take-Grant model (TG) basic concepts. TG is an access control protection model which represents transformation of rights and information between entities inside a system implicitly or explicitly. But it doesn't provide a method to restrict the ACPs with constraints. Also, it doesn't support policy combination and nested policy specification. Unlike TG, CPG provides the means for nested definition and constrained specification of ACPs. Furthermore, CPG contains the necessary rules for the policy combination and expansion. We define the constraints by restricting subjects, objects and actions. In addition, CPG model has the ability of time-dependent policy definition. Different kinds of constraints can be combined by using present method clearly. Furthermore, we can discover the implicit transfers of rights and information by applying a set of rules on each CPG graph. Ultimately, by using CPG model, we can analyze the presence or absence of the conflict or contradiction among ACPs.

Because of the earlier properties, the CPG model is suitable for modeling and analyzing web services ACPs, obtaining the policy of the composite web service and analyzing this outcome. An important problem in web service composition is that partners which contribute in a combination may have inconsistent policies. This makes the composed web service invalid. A few numbers of studies such as Rouached and Claude (2006) and Derakhshandeh *et al.* (2007) have considered this problem. Present method has the capability of solving such problems in the specification, combination and conflict discovery in ACPs.

Corresponding Author: Zahra Derakhshandeh, Department of Computer Engineering, University of Isfahan, Isfahan, Iran
Tel: + 98 (0) 311 7934069 Fax: + 98 (0) 311 6684435

There are some study (Bonatti *et al.*, 2000; Siewe *et al.*, 2003; Jajodia *et al.*, 2001) that consider the combination requirement of ACPs but present model is advantageous than these studies in several issues: First, present model has a more natural and sufficient constraint specification as well as constraint combination (e.g., in modeling temporal constraints) method. Second, it enables us to pose constraints on each three parts of an ACP (i.e., subjects, objects and actions). Third, the CPG model considers derivation of new implicit information and rights transfer in analysis and combination of policies.

TAKE-GRANT (TG) PROTECTION MODEL

The Take-Grant protection model is a formal access control model, which represents transformation of rights and information between entities in a system. It was first presented by Jones *et al.* (1976) to solve the safety problem. In this study, the protection state is represented as a directed finite graph in which vertices are entities of system and edges are labeled. Each label indicates the rights that the source vertex of the corresponding edge has over destination vertex. Entities could be subjects (represented by ●), objects (represented by ○) or play the both roles (represented by ⊗). The set of basic access rights is denoted as R={t, g, r, w} which t, g, r and w, respectively stand for take, grant, read and write access rights. To model the rights transfer, TG uses a set of rules called de-jure rules. These rules transfer TG graph to a new state reflecting the modification of protection state in an actual system. They are used for modeling the transfer of rights in the system. The de-jure rules are take, grant, create and remove (Bishop, 1994, 1995, 1996). There are also another set of rules called de-facto rules. They are used for modeling the transfer of information in the system. The de-facto rules are post, spy, find and pass (Bishop, 1979, 1996). Some of these rules are described briefly as follow:

- **Spy:** Let x, y and z be three distinct vertices in a protection graph G0 and let x and y be subjects. Let there be an edge from x to y labeled α where, $r \in \alpha$ and an edge from y to z labeled β, where, $r \in \beta$. Then the Spy rule defines a new graph G1 with an implicit edge from x to z labeled r (Fig. 1)
- **Find:** Let x, y and z be three distinct vertices in a graph G_0 and let y and z be subjects. Let there be an edge from y to x labeled α where, $w \in \alpha$ and an edge from z to y labeled β, where, $w \in \beta$. Then the find rule defines a new graph G_1 with an implicit edge from x to z labeled r (Fig. 2)

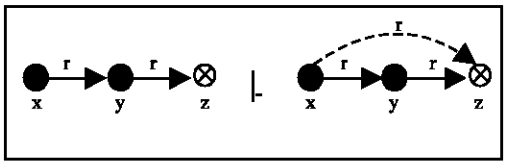

Fig. 1: The spy rule

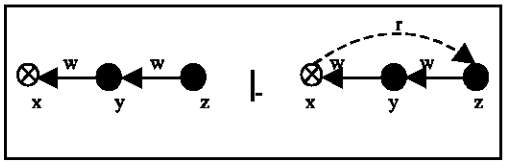

Fig. 2: The find rule

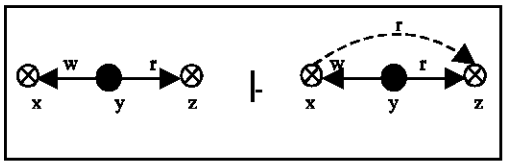

Fig. 3: The pass rule

- **Pass:** Let x, y and z be three distinct vertices in a graph G_0 and let y be a subject. Let there be an edge from y to x labeled α where, $w \in \alpha$ and an edge from y to z labeled β, where, $r \in \beta$. Then the pass rule defines a new graph G_1 with an implicit edge from x to z labeled r (Fig. 3)

CONSTRAINED POLICY GRAPH ACCESS CONTROL MODEL

Any language for expressing ACPs should provide a method for explaining different forms of accesses. ACPs are expressed by subjects, objects and actions. Subjects can be users, groups, roles or applications and objects can be files, documents, or also any subjects. An ACP determines whether a subject is permitted to perform a set of actions on an object or not.

We suggest a method for modeling ACPs based on the TG model. As we saw, TG can model each policy separately and has no means for combining ACPs. More important point is that granting or denying the rights in TG is done unconditionally; while in most cases it is necessary to consider and express the policies depend on some conditions or events (e.g., conditions on time, location, etc.). Di Vimercati *et al.* (2005) is one of the studies discussed the main features of ACPs, that mentioned the need for considering different constraints in the specification of ACPs.

We represent a model that considers these requirements besides the valuable features inherited from TG model such as basic concept of transferring rules.

Model specification: We pointed out the importance of considering constraints in the ACP specification. In present access control model, the applied constraints on the system can influence a subject, an object or an action according to the system requirements. In the other words.

If a subject has a special role, it is possible to have the ability of performing an action towards an object.

A subject has a right (a set of rights) only on specific parts of an object (e.g., a student has only the right of reading digital documents among all library documents).

A subject has a right towards the related object only for performing a specific operation (specific purpose). It is very useful in modeling the web services policies. For example, a user has a specific right on database of a web service (e.g., read and write) only for invoking a special operation of web service while the right of the user towards database, may change for other purposes (i.e., different operations).

On the other hand, an important requirement, common to many applications, is related to the temporal properties of access permissions. In such applications, permissions are granted only for specific periods of time (e.g., periodic authorizations for optimizing resource usage). So, the access control model must have the ability of defining temporal constraints in its ACP specification.

Now we introduce some primitive definitions required for expressing present proposed model.

Definition 1: Basic rights (R): R is the primitive set of the access rights which is considered in the system (e.g., the principal rights of TG model).

Definition 2: Constrained Policy Graph (CPG): If S and O be the set of subjects and objects, respectively, a CPG Ψ is the pair (V, L), where:

- $V \subseteq S \cup O$ is a set of vertices
- L is a set of edges where, $L \subseteq V \times V \times$label and Label = C:P (read P condition to C) is label of the related edge, in which:

  - P is either a set of basic rights ($P \subseteq R$) or is a CPG defined recursively and
  - C is the required constraints for possessing P by the source vertex (subject) of the corresponding edge towards the destination vertex (object). It is defined as the quadruple (C_s, C_o, C_α, C_θ) in which:

    - C_s and C_o are the constraints over subject and object, respectively
    - C_α determines the operations in which P is valid by subject towards object
    - C_θ indicates a set of temporal constraints

Now, we will explain the components of the set C and the way of their definition more precisely:

- **C_s:** It is a constraint on the subject, i.e., only if the subject follows this constraint, its rights (that labeled on the related edge) are valid towards the object. In its basic form, C_s is a simple predicate related to the subject. We use an obligatory parameter for showing its related subject. C_s can be a logical formula in general. If a subject possesses a right towards the object without any constraint on that subject, C_s will be true (T) and if there is no constraint leading the subject to have a right towards the object, C_s will be false (F). The following grammar defines C_s:

$$C_s = C_s \wedge C_s \mid C_s \vee C_s \mid \neg C_s \mid \beta \mid T \mid F$$

β is a simple predicate which is expressed depending on the related system requirements.

- **C_o:** Subject accesses towards the objects can be restricted depending on the content of the related objects. This restriction can be achieved by the conditions (constraints) called content-dependent conditions. We will show these conditions by using C_o when a subject posses a right towards the specific part of the object. Like C_s, C_o is a simple predicate (with an obligatory parameter showing its corresponding object) in its basic form and a logical formula in general. We have:

$$C_o = C_o \wedge C_o \mid C_o \vee C_o \mid \neg C_o \mid \beta \mid T \mid F$$

β is a simple predicate which is expressed depending on the related system requirements.

- **C_α:** A subject may possess different rights on the object based on performing different operation on that object. C_α determines these operations which cause having different rights for the subject towards the object. This constraint is a logical formula as well. Each predicate has two parameters (s and o) which point out the related subject and object of the corresponding edge, respectively. Grammar of C_α is defined as grammar of C_s and C_o, too.

An example of using C_α is in the modeling web services policies, i.e., depends on invoking different methods (operations) of a web service by a subject, this subject has different rights towards the related web service.

- **C_θ:** This constraint, in its basic form, expresses an interval (t_s, t_d) and it means that the mentioned right is valid in the interval between t_s and t_d (an interval is an ordered set of temporal points in which $t_s \leq t_d$). C_θ can be extended to a combination of intervals in its general form. The grammar of C_θ is as follows:

$$C_\theta = C_\theta \wedge C_\theta \mid C_\theta \vee C_\theta \mid \neg C_\theta \mid \beta \mid T \mid F$$

β is a time interval in the form of (t_s, t_d), T means always and F means never (no time).

Obviously if any constraint in quadruple C be False (F), then C is False (the related label is F: P) means never and if all constraints of C are True (T), then C equals True which means unconditionally or without any constraint. In this state the label of related edge is T: P or simply P.

In Definition 2, P, in its basic form, is the set of basic rights defined in the system. But we have extended it to be a policy again because of this reason: Sometimes instead of possessing a right towards an object by a subject (i.e., $P \subseteq R$), the subject may follow a policy for accessing the object. So, P is another CPG (another policy). For example, a college should follow the public policy of library for accessing the library documents. Therefore, we should provide the possibility of nested policy definition for modeling these situations.

Assume that P and Q are two CPGs and the source vertex v_a in policy Q has the policy P towards the destination vertex v_b in the condition C. Thus, v_a corresponds to one of the present vertices in CPG P (we call this vertex D). Therefore, all policies and rights that D has over other entities are also applied on v_a. Also, if vertex v_a itself contains other entities, they will follow the new changes recursively (example 3).In fact, we encounter with label C: P_D for the connecting edge of two vertices v_a and v_b.

For changing a graph with such an edge into a basic graph (i.e., a graph with P from the set R) and studying how to transfer rights and information, we define the policy expansion procedure as follows:

Definition 3: Policy expansion: If $Q = (V_1, L_1)$ be a CPG in which $l_1 = (v_a, v_b, C: P_D) \in L_1$ and $P = (V_2, L_2)$ be another CPG that $l_2=(D, v_b, C': P'_X) \in L_2$ and assume that $L'_2 \subseteq L_2$ be the set of all edges outgoing from the vertex D, then $K = (V, L)$ is the expanded CPG of Q regarding P:

$V = V_1 \cup V_2$,
$L = (L_1 - \{l_1\}) \cup (L_2 - L'_2) \cup \{ l'_i \mid l'_i = (D, v_i, C \wedge C'_i : P'_{iX}), (D, v_i, C'_i : P'_{iX}) \in L_2, i = 1, 2, \ldots \}$,
$V_a = D$

The later expression means that vertices D and v_a are merged together. This procedure will continue (say in n steps) until all the nested constraint policy graphs expanded to the basic rights (R). In this state the final edge between v_a and v_b will be $l= (v_a, v_b, C \wedge C' \wedge \ldots \wedge C_{(n)}: R)$.

Definition 4: Combining constraints: If $C_1 = (C_{1s}, C_{1o}, C_{1\alpha}, C_{1\theta})$ and $C_2= (C_{2s}, C_{2o}, C_{2\alpha}, C_{2\theta})$, then:

$C_1 \wedge C_2 = (C_{1s} \wedge C_{2s}, C_{1o} \wedge C_{2o}, C_{1\alpha} \wedge C_{2\alpha}, C_{1\theta} \wedge C_{2\theta})$,
$C_1 \vee C_2 = (C_{1s} \vee C_{2s}, C_{1o} \vee C_{2o}, C_{1\alpha} \vee C_{2\alpha}, C_{1\theta} \vee C_{2\theta})$,
$\neg C_1 = (\neg C_{1s}, \neg C_{1o}, \neg C_{1\alpha}, \neg C_{1\theta})$,

In which:

$C_{1\theta} \wedge C_{2\theta} = (t_{1s}, t_{1d}) \wedge (t_{2s}, t_{2d}) = \{t \mid (t \in (t_{1s}, t_{1d})) \wedge (t \in (t_{2s}, t_{2d}))\}$
$C_{1\theta} \vee C_{2\theta} = (t_{1s}, t_{1d}) \vee (t_{2s}, t_{2d}) = \{t \mid (t \in (t_{1s}, t_{1d})) \vee (t \in (t_{2s}, t_{2d}))\}$
$\neg C_{1\theta} = \neg (t_s, t_d) = \{t \mid (t < t_s) \wedge (t > t_d)\}$

Example 1: A college must follow the policy of university library in accessing to library documents and the policy of library contains the following rules:

- If the applicant be guest of library, he/she is allowed to read the documents only at the periodical time t that $t_{s\_free} \leq t \leq t_{d\_free}$. So, the related condition is: C_1=(guest(s), T, T, (t_{s_free}, t_{d_free}))
- If the applicant is one of the university teachers, he/she is allowed to read and change the documents, i.e., C_2= (teacher(s), T, T, T)
- If the applicant is a student, he/she can read documents, so C_3=(student(s), T, T, T)

Figure 4 shows the whole library policy by using CPG. We see that an applicant depends on his/her role has different rights over documents.

Now, assume the policy of the university college is partially as follow:

- The college obeys the library policy in autumn and winter, i.e., $C'1 = (T, T, T, (t_{s\_aw}, t_{d\_aw}))$

Because the college has the role as an applicant, it must be mapped to applicant vertex of library policy (Fig. 5). Now, we use policy expansion procedure for expanding the college policy over library documents to reach the basic CPG (Fig. 6). For instance the result of $C'_1 \wedge C_1$ is as follows:

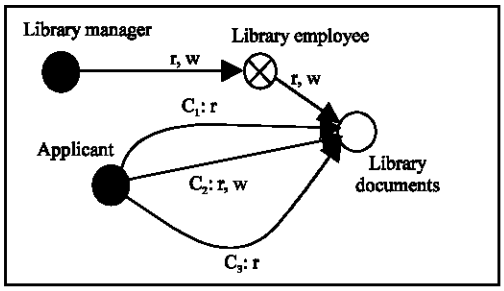

Fig. 4: The policy of library

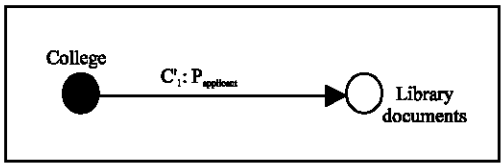

Fig. 5: The policy of college

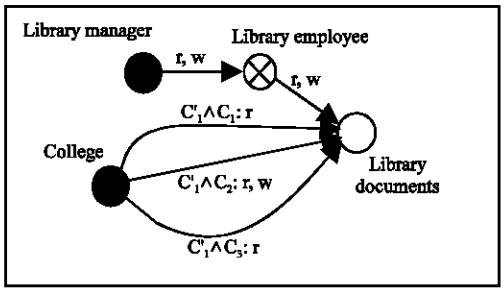

Fig. 6: The college policy after expansion

$C'_1 \wedge C_1$= (guest(s), T, T, [t_{s_free}, t_{d_free}]) ∧(T, T, T, [t_{s_aw}, t_{d_aw}])= guest(s), T, T, ([t_{s_free}, t_{d_free}]∧[t_{s_aw}, t_{d_aw}]))

At the same time, it is possible that the college itself contains other entities which are connected to each other according to their related policies. Assume that the college has manager, student and teacher so its policy is a combination of these partial policies.

Combining ACPs: Although different organizations protect their resources, they need to be connected together to reach their common complicated goals. For example, nowadays composition of web services is one of the key requirements for responding to the requirements of users whose requests don't meet by existing web services (Charfi and Mezini, 2005). Moreover, verifying that whether the composed web service supports the ACPs of participant web services or not, is a challenge. Few studies such as (Rouached and Claude, 2006; Derakhshandeh *et al*., 2007) have ever tried to represent a solution for this problem.

The following grammar expresses kinds of present proposed policy combinations:

$$K = P \cup Q \mid P \cap Q \mid P\text{-}Q \mid \neg P$$

where, K, P and Q are CPGs.

Union (P∪Q): By means of this, two policies are merged into a policy which contains a union of both policies. This operator admits the access which one or both of its components permit.

Definition 5: Union (P∪Q): If $P = (V_1, L_1)$ and $Q = (V_2, L_2)$ are two CPGs, then K = (V, L) is the union of these two graphs, in which $V = V_1 \cup V_2$ and $L = L_1 \cup L_2$.

Note that if a vertex is a subject in one graph and is an object in another one, then it plays the both roles (i.e., ⊗) in the graph of union.

Sometimes after union of two CPGs, we can simplify the combined graph. The following definition shows such a situations and the way of simplifying the graph.

Definition 6: CPG simplification: If P = (V, L) be a CPG in which $l_1 = (v_a, v_b, C_a: P_1) \in L$ and $l_2 = (v_a, v_b, C_b: P_2) \in L$, also there is a unifier θ that $\theta(C_a) = \theta(C_b) = C$, then we have $l = (v_a, v_b, C: P_1 \cup P_2) \in L$ where:

- If $P_1, P_2 \in R$, then $P_1 \cup P_2$ is the union of these two sets
- If P_1, P_2 are two CPGs, their union is produced according to definition 5 and then the policy expansion procedure could be applied on the produced graph to reach the basic CPG. Note that we can, at first, expand the nested graphs by applying the policy expansion procedure and then get the union of both produced graphs as definition 5
- If either P_1 or P_2 is a member of R (∈R) and another one is a CPG, at first we expand the nested graph and then get the union of two graphs as definition 5 again

Example 2: Consider the college in example 1 with manager of the college, teacher and student. Suppose that each one has its own policy. For example consider that the policy of manager is as follows:

- The manager is allowed to read the information of the teachers for verifying them: C''_1= (T, T, verify(s, o), T)
- He/she can change this information at the time of selecting the teachers for instructing lessons: C''_2 = (T, T, T, [$t_{s\_selection}$, $t_{d\_selection}$])

Figure 7 shows the policy of the college manager.

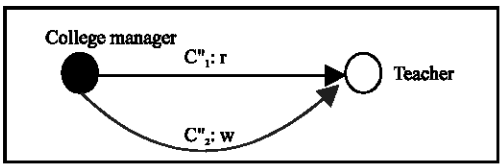

Fig. 7: The policy of the college manager

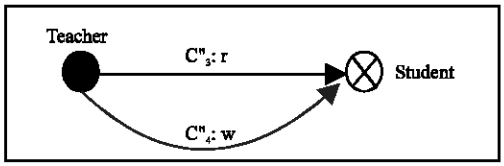

Fig. 8: The teacher policy

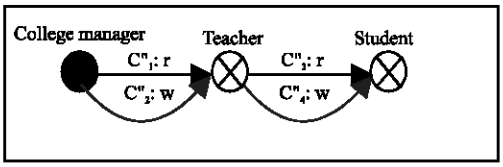

Fig. 9: The college policy

On the other hand, the policy of the teacher is as follows:

- Teacher is allowed to read the students information at the exam season: $C"_3 = (T, T, T, [t_{s\_exam}, t_{d\_exam}])$
- He/she can change the student information at the time of marks announcement: $C"_4 = (T, T, T, [t_{s\_announcement}, t_{d\_announcement}])$

The teacher policy is shown in Fig. 8.

Now, we obtain the college policy based on its characters policies. It is gained from the union of their policies (Fig. 9). Note that the teacher vertex changes from object state into subject-object state.

Example 3: As we saw, Fig. 6 was obtained from applying the policy expansion procedure on the college policy. On the other hand, the college contains mentioned characters. So, they must obey the college policy towards the library documents in order to access them. Figure 10 shows the college policy towards library documents considering its characters. We only show the teacher policy towards documents for sake of figure clarity, i.e., the policies of college manager and student towards documents were omitted to keep the clarity of the Fig. 10.

Similarly, the university as a subject must follow the policy of its upper organization e.g,. Department of Research in its educational rules. Furthermore, the university consists of many entities, with their own policies, that may be interrelated together. By using mentioned method, the university policy can be obtained based on its components policies and relations.

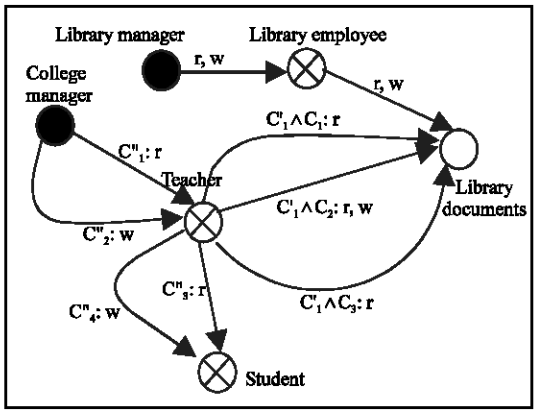

Fig. 10: College policy towards documents (considering its principals)

Present model can express the groups of users who follow a common policy. It is possible to apply a policy for all group members or except some members of this group with some constraints (i.e., if these members obey the constraints, they have similar policy as other members of that group).

On the other hand, since system entities have various rights towards each other and because of their relations; it is possible to create new rights or information flow in the system. Here, we mentioned these transfers. These implicit transfers are not considered in many ACPs modeling methods. Without considering indirect transfers, the combination results and analysis outcomes may be incomplete or even incorrect. Therefore, we utilized TG transferring rules and completed them according to present goals. Present method for applying transferring rules is as follows:

CPG transferring rules: As mentioned before, applying transferring rules on CPGs results in new edges from two existing ones, such as TG model. But the transfer occurs when two constraint sets of related existing edges, contributed in resulting new edge, occur simultaneously. Also, each parameter of constraints sets is mapped to related vertex during the new transfer (i.e., each Cs parameter is mapped to related subject and each Co parameter is mapped to related object and each Cα parameter is mapped to related subject and object). For example, we show this method for spy rule in Fig. 11.

- **Spy:** Let x, y and z be three distinct vertices in a CPG_0 and let x and y be subjects. Let there be an edge from

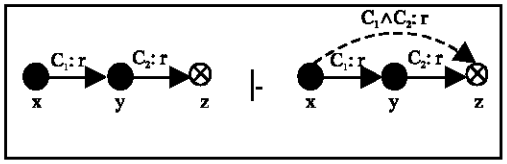

Fig. 11: The spy rule

x to y labeled C_1: r and an edge from y to z labeled C_2: r. Then the spy rule defines a new graph CPG_1 with an implicit edge from x to z labeled $C_1 \wedge C_2$: r (Fig. 2)

After this, present intention of transferring rules is CPG transferring rules.

Notice that policy expansion procedure and transferring rules both change the CPGs from one state into another state. So, it is necessary to apply both of them (if possible) before combining or analyzing the ACPs. It is necessary to obtain correct results for at least some kinds of combinations e.g., Intersection, Subtraction, etc.

Intersection (P ∩ Q): It allows only those accesses that are permitted by both components P and Q. For example, suppose that the library employee allows the college students to access to the library documents. On the other hand, the manager of college permits specific persons to access to these documents. In this case, ACP of college students to the mentioned documents is obtained from the intersection of both policies (i.e., only the accesses are valid which both manager and library employee permit them). The intersection CPG is obtained as follows:

- If nested policy exists in P or Q, we must apply the policy expansion procedure to reach the basic CPG for each one (P or Q)
- Then, we apply CPG transferring rules to add the implicit transfers to each CPG, if exist
- Finally, we intersect two resulted CPGs as follow:

Definition 7: Intersection (P∩Q): If $P = (V_1, L_1)$ and $Q = (V_2, L_2)$ are two CPGs, then $K = (V, L)$ is the intersection of these two graphs in which $V = V_1 \cap V_2$ and for each two edges $l_1 = (v_a, v_b, C_1: P_1) \in L_1$ and $l_2 = (v_a, v_b, C_2: P_2) \in L_2$, the edge $l \in L$ is the corresponding edge of v_a and v_b in K that $l = (v_a, v_b, C_1 \wedge C_2: P_1 \cap P_2)$, where:

- $P_1, P_2 \in R$ then $P_1 \cap P_2$ is the intersection of them. If the resulted set of $P_1 \cap P_2$ has no member, the related edge is omitted

Subtraction (P-Q): It restricts policy P by eliminating all the accesses in a second policy (Q). The subtraction graph is obtained as follows:

- If the nested policy exists in P or Q, we apply the policy expansion procedure to reach the basic CPG for each one (P or Q)
- Then, we apply transferring rules on each CPG to obtain implicit transfers, if exist
- Finally, we subtract two resulted CPGs as follows:

Definition 8: Subtraction (P-Q): If $P = (V_1, L_1)$ and $Q = (V_2, L_2)$ are two CPGs and if $L'_1 \subseteq L_1$ is a set of edges in P, which have the same corresponding beginning (subject) and end (object) vertices in both P and Q (i.e., every $l_1 = (v_a, v_b, C_1: P_1) \in L_1$ in P which has an equivalent edge $l_2 = (v_a, v_b, C_2: P_2) \in L_2$ in Q, is placed in L'_1 set), then $K = (V, L)$ is the subtraction of these two graphs that:

$$V = V_1, \quad L = (L_1 - L'_1) \cup \{l_i \mid l_i = \begin{cases} (v_a, v_b, C_{1i} \wedge C_{2i}: P_{1i} - P_{2i}) & \text{if } \theta(C_{1i}) = \theta(C_{2i}) \\ (v_a, v_b, C_{1i}: P_{1i}) & \text{otherwise} \end{cases}$$

Where:
$P_1, P_2 \in R$, so P_1-P_2 is the subtraction of them.

Negation (¬ Q): A CPG expresses the rights of its subjects towards related objects based on constraints. Negation of the CPG expresses possessing negative of mentioned rights for related subject towards the object in the same constraints. Thus:

Definition 9: Negation (¬Q): If $Q = (V, L)$ be a CPG and $L = \{l_i \mid l_i = (v_{ai}, v_{bi}, C_i: P_i)\}$, then $K = (V', L')$ is the negation of Q in which:

$$V' = V, L' = \{l'_i \mid l'_i = (v_{ai}, v_{bi}, C_i: \neg P_i)\}$$

Where:

- If $P \in R$, $\neg P$ is the negation of this set
- If P is a CPG, then $\neg P$ will be obtained recursively by using definition 9

In addition to mentioned operators we can model conditional relations by using CPGs as follows:

Conditional relations: The subject v_a, in condition C, possesses the policy P towards other entity (v_b) and for the other conditions (except C), it has the policy Q towards v_b. In fact, we encounter with conditional expression: if C then P else Q for v_a towards v_b. It can be modeled by using a CPG, as Fig. 12.
$\neg C$ is obtained by using definition 4.

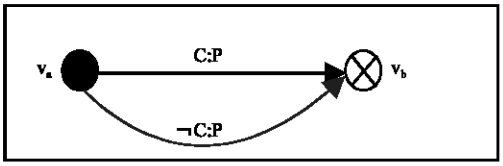

Fig. 12: Conditional relations CPG model

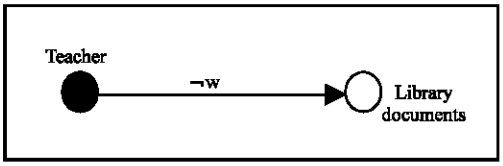

Fig. 13: A part of university policy

Conflict detection: During combination of CPGs or verification of them, it is possible to encounter with a case in which a subject simultaneously has two inconsistent policies or rights towards a same object. In fact, the subject is allowed by a policy to do an action on an object and simultaneously is prevented by other policy from doing the same action on that object. We say the system was encountered with conflict.

Definition 10: Conflict: If Q = (V, L) be a CPG in which $v_a, v_b \in V$ and $l_1 = (v_a, v_b, C_a: P) \in L$ and $l_2 = (v_a, v_b, C_b: \neg P) \in L$ and also there is a unifier θ that $\theta(C_a) = \theta(C_b) = Ce$, then the corresponding edges of these two vertices v_a and v_b will be $l_1 = (v_a, v_b, C_e: P)$ and $l_2 = (v_a, v_b, C_e: \neg P)$, which means conflict.

Example 4: Assume a part of university policy contains the following rules:

- A teacher is not allowed to change the university documents such as library documents and educational documents.

The university considers this policy unconditionally (i.e., all parts of condition set are true (T)) (Fig. 13).

Meanwhile according to college policy, the teacher, as one of the university characters, has r and w rights on library documents based on constraint $C'_1 \wedge C_2$ (example 3, Fig. 10). We again show the part of college policy about teacher and library documents in Fig. 14.

The university policy is obtained as combination of its entities policies (e.g., colleges, etc). Like before, two vertices teacher and library documents are common in both policies and the corresponding edge of them in condition $C'_1 \wedge C_2$ make the system encounters with the conflict (it is resulted from two edges l = (teacher, library documents, ¬w) and l' = (teacher, library documents, $C'_1 \wedge C_2$: r, w) (Fig. 15). It means that in condition $C'_1 \wedge C_2$ a teacher is allowed to change the library documents and also is prevented from changing them.

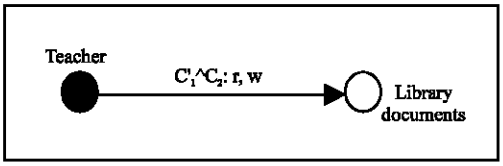

Fig. 14: The teacher policy on library documents

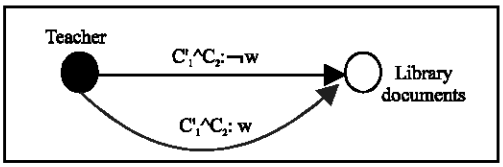

Fig. 15: Conflict example

MORE ANALYSIS OF ACPs

Here, we define a method for more accurate analysis of the ACPs according to CPG model features. By using this model not only we are able to specify the ACPs, but also we can examine transferring of rights and information in related systems.

In many cases, the system works according to integration of system parts, i.e., it works based on a combined process obtained from combination of system parts processes. While the resulted process of the combination seems faultless, but after obtaining the policy of combined process and analyzing it, we may encounter with some inconsistencies. For example the resulted policy of combination has contradictions with the policy of each participated parts. Or although the policy of one part is in contradiction with some others, this part is used in the combination process by mistake (e.g., in web service composition as mentioned before).

In general, analysis of one model is arising from responses given to questions which are posed in mind. What is primarily important in an access control model is how to transfer the rights and information in related system. More accurately, we are interested in answering the following questions:

- Based on providing which condition (conditions), the subject x obtains the right α towards the object y?
- Can subject x obtain the right α based on occurring condition C towards the object y?

Since CPG model considers the constraints in modeling the ACPs, it is possible to consider and analyze constrained transfers accurately. The answer of the first question is very useful in many cases (e.g., the constraint resulted from this question can be analyzed and adjusted more accurately and etc). The second question distinguishes whether a transfer occurs based on distinct conditions or not.

The TG model has defined predicates for analyzing the graphs (Bishop, 1979, 1994, 1995, 1996). We mention the can.know predicate as an example:

- **Can.know(x,y,G_0):** The predicate can.know(x, y, G_0) is true if and only if there exists a sequence of protection graphs G_0, …, G_n such that G_n is derived from G_0 by rule applications and in G_n there is an edge from x to y labeled r or an edge from y to x labeled w and if the edge is explicit, its source is a subject

Since TG predicates are defined unconditionally, they can be used for analyzing the CPGs in always-true conditions. But for our interests, we define following predicate based on proposed model.

Definition 11: The predicate can.obtain (C, α, x, y, CPG_0) is true for the condition C, the access right α and vertices x (as a subject) and y (as an object or a subject), if and only if there exist the graphs CPG_0, CPG_1,…, CPG_n, so that CPG0|-*CPG_n by applying transferring rule and there is an edge with label C: α from x to y in CPG_n.

If we assume the set of all desired rules as the set of A, transferring from CPG_0 to CPG_n, is resulted from applying whole applicable rules on CPG_i ($0 \le i \le n-1$), beginning from CPG_0, with this condition that after obtaining CPG_n, the possibility of applying any rules of A on CPG_n doesn't exist. Of course this process can be ended when desired result was found (when the desired edge with label of α appears from x to y) otherwise it will continue until reaching CPG_n.

Example 5: Remember the graph of Fig. 9 from example 2. Suppose we want to know whether the manager can obtain the right r towards the student or not (If yes, in which constraint). In other words, is the result of predicate can.obtain(C, r, manager, student, CPG_0) true? In which constraint (C)?

After applying the spy rule, Fig.16 is obtained. Two edges l_1= (manager, teacher, C"; r) and l_2 (teacher, student, C"$_3$: r) contribute to create the implicit edge. As mentioned before, the parameters of constraints C"$_1$ and C"$_3$ are mapped to related vertices and the conjunction of

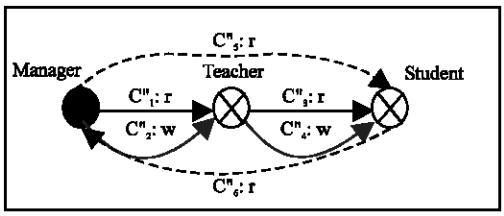

Fig. 16: Applying spy and find rules on college graph

them results the constraint of new edge. So, for the new edge l = (manager, student, C"$_5$: r), the constraint C"$_5$ is obtained as follows:

$C"_5 = C"_1 \wedge C"_3 = (T, T, verify(manager, teacher), T) \wedge (T, T, T, [t_{s\_exam}, t_{d\_exam}]) = (T, T, verify(manager, teacher), [t_{s\_exam}, t_{d\_exam}])$

Also, after applying the Find rule, the edge l'= (student, manager, C½$_6$: r) is created and C½6 is:

$C"_6 = C"_2 \wedge C"_4 = (T, T, T, ([t_{s\_selection}, t_{d\_selection}] \wedge [t_{s\_announcement}, t_{d\_announcement}]))$

As we saw, the transfer of information has two different directions depends on system constraints (the constraints which are held in the system). If condition C"$_5$ holds, the manager of college has the right of reading the student information in the combined graph and with holding condition C"$_6$, the student reads the manager information. These transfers occur after combining the manager policy and the teacher policy for obtaining the college policy.

RELATED WORKS

As we described earlier, there are many methods about modeling ACPs. Many old methods did not consider the possibility of policy combination. For example in (Bertino *et al.*, 1996, 1998) a temporal model for access control has been proposed. A time interval is associated with each authorization to determine the period of time in which the authorization holds. However their framework cannot handle the enforcement of multiple policies. But there are some new works that consider the combination requirement of ACPs. We review some cases as follows:

In Bonatti *et al.* (1999), an algebra method has been presented for specifying and combining the ACPs. While present method is similar to this method in basic concepts, we also consider the temporal constraints in expressing ACPs. Furthermore, our method has a more natural constraint specification as well as constraint combination.

Since system entities have various rights towards each other and because of their relations, it is possible to appear new flow of rights or information in the system. We discover these rights and information transfers (if exist) in the system and add them to the policy graph of the system by applying some rewriting rules. Therefore what's obtained from all kinds of the combinations is exactly based on real transfers of system rights and information (either explicitly or implicitly). Note that without considering transfers which is appeared indirectly in the system, the combination results may be incomplete or even incorrect. Bonatti *et al.* (2000) didn't consider this fact, too.

In Jajodia *et al.* (2001), a method has been expressed for applying multiple ACPs in a system. In this study, different policies are specified by a unique language and they could be analyzed with a set of defined decision rules. These policies are saved in the system library and the new rules are added to the library based on the users' requests. In this course, applying several policies in the system means translating the policies to a unit language and saving and analyzing them. We consider the different kinds of policy combinations on demand. Furthermore, this method has only considered the constraints for classifying the objects and grouping the subjects. Also it hasn't considered the access permissions which are valid at a time period (temporal constraints).

Siewe *et al.* (2003) represents a method for specifying ACPs and combining them. The method considers the policies according to the temporal relationships among the access permissions and expresses them based on the Interval Temporal Logic (ITL). But, as we explained in section 3, an access control specification method needs to consider other constraints in addition to the temporal ones (e.g., constraints for restricting subjects, objects, etc), for specifying all kinds of ACPs. Such constraints have not been studied in (Siewe *et al.*, 2003).

Other benefit of present proposed model is in visualizing graphs which have the higher level of understanding compared to algebra and pure logic methods.

CONCLUSION AND FUTURE WORKS

In this study, we presented a method for expression and combination of ACPs. This method is capable of expressing nested ACPs, conditional (constrained) ACPs, combination of ACPs and verification of them for finding conflict or contradiction among the combined policies. We have provided the possibility of expressing the groups of users and defining the policy for all the members of the group. Simultaneously, we can provide a hierarchy for restricting a member (or some members) of the group to a special policy or constraints. This method enjoys the high understanding ability of the graphs by their graphical schema besides possessing a logical infrastructure. Furthermore, we consider the implicit transfers of the rights and information happenings because of the entities relations with different policies in the system as well as explicit ones. It also can be used for obtaining precise combination of the ACPs.

The proposed method could be adopted easily to be used in the web service composition for specification of ACPs in composing web services and analyzing the composed web service. We are trying to show the concept by introducing a number of case studies in this regard.

REFERENCES

Bertino, E., C. Bettini, E. Ferrari and P. Samarati, 1996. A temporal access control mechanism for database systems. IEEE Trans. Knowledge Data Eng., 8: 67-80.

Bertino, E., C. Bettini, E. Ferrari and P. Samarati, 1998. An access control model supporting periodicity constraints and temporal reasoning. ACM Trans. Database Syst., 23: 213-285.

Bertino, E., S. Jajodia and P. Samarati, 1999. A flexible authorization mechanism for relational data management systems. ACM Trans. Inform. Syst. 17: 101-140.

Bishop, M., 1979. The transfer of information and authority in a protection system. Proceeding of the 7th ACM Sym. Oper. System Principles, December 10-12, Pacific Grove, California, United States, pp: 45-54.

Bishop, M., 1994/1995. Theft of information in the take-grant protection model. J. Comput. Sec., 3: 283-309.

Bishop, M., 1996. Conspiracy and information flow in the take-grant protection model. J. Comput. Sec., 4: 331-360.

Bonatti, P., S.D.C. di Vimercati and P. Samarati, 2000. A modular approach to composing access control policies. Proceeding of the 7th ACM Conference Communications Security, November 01-04, ACM New York, USA., pp: 164-173.

Charfi, A. and M. Mezini, 2005. Middleware Services for Web Service Compositions. 1st Edn., Software Technology Group Darmstadt University of Technology, Chiba, Japan.

Derakhshandeh, Z., B.T. Ladani and N. Nematbakhsh, 2007. Verification of access control policies in composition of web services using take-grant protection model. Proceeding of the 4th Iranian Soc. Cryp. Conference (ISCC07), October 16-18, Iran University of Science and Technology, pp: 185-185.

Di Vimercati, S.D.C., P. Samarati and S. Jajodia, 2005. Policies, models and languages for access control. Databases in Networked Information Systems: 4th International Workshop, LNCS., 3433, March 2005, Springer-Verlag, pp: 225-237.

Jajodia, S., P. Samarati, V.S. Subrahmanian and E. Bertino, 2001. A unified framework for enforcing multiple access control policies. ACM Trans. Database Syst., 26: 214-260.

Jones, A.K., R.J.S. Lipton and Lawrence, 1976. A linear time algorithm for deciding security. 17th Annual IEEE Sym. Found. Comp. Science (FOCS), October 1976, IEEE Xplore Pub., pp: 33-41.

Rouached, M. and G. Claude, 2006. Securing web service compositions: Formalizing authorization policies using event calculus. Int. Conf. Service-Oriented Comput., 4294: 440-446.

Siewe, F., A. Cad and H. Zedan, 2003. A compositional framework for access control policies enforcement. Proceedings of the 2003 ACM Workshop on Formal Methods in Security Engineering (FMSE'03), October 30, Washington, DC. USA., pp: 32-42.