Specification and Enforcement of Static Separation-of-Duty Policies in Usage Control[*]

Jianfeng Lu, Ruixuan Li[**], Zhengding Lu, Jinwei Hu, and Xiaopu Ma

Intelligent and Distributed Computing Lab, College of Computer Sci. and Tech., Huazhong University of Sci. and Tech., Wuhan 430074, P.R. China
lujianfeng@smail.hust.edu.cn, {rxli,zdlu,jwhu}@hust.edu.cn, xpma@smail.hust.edu.cn

Abstract. Separation-of-Duty (SoD) policy is a fundamental security principle for prevention of fraud and errors in computer security. The research of static SoD (SSoD) policy in recently presented usage control (UCON) model has not been explored. Consequently, this paper attempts to address two important issues: the specification and enforcement of SSoD in UCON. We give a set-based specification scheme, which is simpler and more general than existing approaches. As for the enforcement, we study the problem of determining whether an SSoD policy is enforceable, and show that directly enforcing an SSoD policy is a coNP-complete problem. In indirect enforcement, we generate the least restrictive static mutually exclusive attribute (SMEA) constraints to enforce SSoD policies, by using the attribute level SSoD requirement as an intermediate step. The results are fundamental to understanding the effectiveness of using constraints to enforce SSoD policies in UCON.

Keywords: Separation-of-Duty, usage control, constraint.

1 Introduction

Separation-of-duty (SoD) is widely considered to be a fundamental security principle for prevention of fraud and errors in computer security, and widely applied in business, industry, and government [1,2]. Although SoD has been studied extensively in the information security, and it has been recognized that "one of RBAC's great advantages is that SoD rules can be implemented in a natural and efficient way" [3], as a related and fundamental problem, research of SoD policy in recently presented usage control (UCON) [4] model has not been explored. UCON has been considered as the next generation access control model with distinguishing properties of decision continuity and attribute mutability [4,5]. Consequently, this paper focuses on static SoD (SSoD) policies in $\mathrm{UCON_A}$

[*] This work is supported by National Natural Science Foundation of China under Grant 60873225, 60773191 and 60403027, National High Technology Research and Development Program of China under Grant 2007AA01Z403.

[**] Corresponding author.

which is a sub-model of UCON only considering authorizations. Since an authorization decision is determined by subject's and object's attributes, and these attribute values can be updated as side-effects of the authorization, the study of SSoD policies in authorization models is more pressing than that in obligation and condition models. In this paper, we provide a set-based specification scheme for SSoD policies. Furthermore, we study a number of problems related to generating SMEA constraints for enforcing SSoD policies in UCON_A systems. We study the problem of determining whether an SSoD policy is enforceable, and generate SMEA constraints to indirect enforce SSoD policies, by using attribute level SSoD requirements (ASSoD) as an intermediate step from SSoD policies to SMEA constraints. The research of the SSoD policy in UCON_A is important for emerging applications as usage control scenarios, and it can also increase UCON's strengths in that it enables the use of constraints to support SSoD policies.

The rest of this paper is organized as follows. Section 2 describes related works. Section 3 gives the specification of SSoD policies. Section 4 studies the problem of determining whether an SSoD policy is enforceable. Section 5 uses SMEA constraints to indirect enforce SSoD policies. Section 6 concludes this paper.

2 Related Work

The concept of SoD can be traced back to 1975 when by Saltzer and Schroeder [6] took it as one of the design principles for protecting information, under the name "separation-of-privilege". The research community has taken an active interest in incorporating SoD controls into computer systems since the late 1980s, Clark and Wilson [1] applied SoD principle to data objects to ensure integrity and to control frauds along with well-formed transactions as two major mechanisms for controlling fraud and error. Later on, SoD has been studied by various researchers as a principle to avoid frauds. One of the best known requirements for SoD is embodied in the Chinese Wall model [7], in which access to documents that could result in a commercial conflict of interest is strictly controlled.

In this paper, the specification scheme of the SSoD policy we propose has its basis in our set-based approach to conflict of interest, and it is considerably simpler syntactically than other schemes because the SSoD policy is expressed in terms of restrictions on permissions other than attributes, such as roles, and we make no attempt to define the conditions that must be met for the constraint to be satisfied. As for the enforcement, Sandhu presented transaction control expressions, a history based mechanism for dynamically enforcing SoD policies [8,9]. Simon and Zurko combined the Object SoD and Operational SoD and introduced a notion of history based SoD [10]. Crampton [11] employed blacklist to enforce historical constraints, it does not need to keep a historical record. However, since these approaches for SoD only consider constraint sets with a few elements, they will have unacceptable overheads to support large range of constraints. UCON_A includes RBAC which increases the difficulty of enforcement

SSoD policies in $\mathrm{UCON_A}$. Motivated by the SMER constraints [12],we enforce SSoD policies in $\mathrm{UCON_A}$ by using SMEA constraints. For a more detailed description of UCON, the reader can refer to [4,5].

3 The Specification of SSoD Policies

We now give a formal basis for representing SSoD policies in $\mathrm{UCON_A}$ models based on the following requirements. (1) An SSoD policy must be a high-level requirement. Clark et al. identified SoD as a high-level mechanism that is "at the heart of fraud and error control" [13]. It states a high-level requirement about the task without the need to refer to individual steps in the task. (2) An SSoD policy must be described in terms of restrictions on permissions. In the ANSI RBAC standard [14], the distinction between SSoD policies as objectives and static mutually exclusive role (SMER) constraints as a mechanism is not clear. One problem is that the SMER constraints may be specified without a clear specification of what objectives they intend to meet; consequently, it is unclear whether the higher-level objectives are met by the constraints or not. Another problem is that even though when SMER constraints are specified there exist a clear understanding of what SSoD policies are desired, when the assignment of permissions to roles changes, the SMER constraints may no longer be adequate for enforcing the desired SSoD policies [12]. (3) An SSoD policy must capture restrictions on user set involved in the task. In practice, the number of users in any organization is bounded. It needs to consider the SSoD policies with an upper bound on the number of users in an access control state.

Definition 1. *An SSoD policy ensures that at least k users from a user set are required to perform a task that requires all these permissions. Formally,*

- *P and U denote the set of permissions, the set of users;*
- *$UP \subseteq U \times P$, a user-permission assignment relation;*
- *$auth\_P_\varepsilon[u] = \{allowed(u,p) \Rightarrow preA(ATT(u),p)\}$;*
- *$\forall(P,U,k) \in SSoD, \forall U' \subseteq U : |U'| < k \Rightarrow \bigcup_{u \in U'} auth\_p_\varepsilon(u) \not\supseteq P$.*

where $P = \{p_1,\ldots,p_m\}$, $U = \{u_1,\ldots,u_n\}$, m, n, and k are integers, such that $2 \leq k \leq min(m,n)$, min returns the smaller value of the two. $ATT(u)$ denotes the user's attributes, preA is the pre-authorizations in UCON, and $allowed(u,p)$ indicates that user u is assigned permission p. ε is a $\mathrm{UCON_A}$ *state which constituted by the set of assignments for all objects' attributes. We write an SSoD policy as $ssod < P,U,k >$. We say that a* $\mathrm{UCON_A}$ *state ε is $safe$ with respect to an SSoD policy e, if in state ε no $k-1$ users from U together have all the permissions in P, and we write it as $safe_e(\varepsilon)$. An* $\mathrm{UCON_A}$ *state ε is $safe$ with respect to a set E of SSoD policies, which we denote by $safe_E(\varepsilon)$, if and only if ε is safe with respect to every policy in the set E.*

4 Enforceability of SSoD Policies

In a $\mathrm{UCON_A}$ system, not all SSoD policies are enforceable. For example, given an SSoD policy $e = \{ssod < \{p_1,p_2\},\{Alice,Bob,Carl\},2>\}$, which ensures that at least two users together from $\{Alice, Bob, Carl\}$ are allowed to have $\{p_1, p_2\}$. Assume that $allowed(u, p_1) \Rightarrow ATT(u) = \{engineer, student, 50\}$, where engineer is a role, student denotes the identity of user, and 50 is a trust value. It means that a user who must cover all these attributes can be allowed to have p_1. Where $allowed(u, p_2) \Rightarrow ATT(u) = \{programmer, student, 75\}$ has the similar meaning. Suppose that $ATT(Alice) = \{supervisor, student, 100\}$, where *supervisor* is a senior role to both *engineer* and *programmer*. Obviously, $safe_e(\varepsilon)$ is false, because *Alice* can be a member of both p_1 and p_2. In order to address this, forbid *Alice* from having the attribute set $\{supervisor, student, 100\}$. This is undesirable, if an attribute can not be assigned to a user, then the attribute value should not be included in the domain of the user attribute.

Definition 2. *(I, M) is a attribute set, where I is the set of immutable attributes, and M is the set of mutable attributes. We say $(I_1, M_1) \leq (I_2, M_2)$ if and only if for each attribute $a \in I_1$, there exists an attribute $a' \in I_2$ such that $a \leq a'$; and for each attribute $b \in M_1$ there exists an attribute $b' \in M_2$ such that $b \leq b'$.*

Obviously, $\leq$ associates the user attribute sets, and these associations form a combined hierarchy that is partially ordered. If (I, M) satisfies exactly the requirement of $allowed(u, p)$, we say (I, M) is the threshold attribute set of p.

Definition 3. *Given an SSoD policy ssod $< P, U, k >$, let (I_{p_i}, M_{p_i}) is the threshold attribute set of each p_i in P, and (I_t, M_t) is an attribute set. If $\forall (I_{p_i}, M_{p_i})$ $(I_{p_i} \leq I_t \Rightarrow M_{p_i} \leq M_t)$, we say (I_t, M_t) is an ancestor attribute set. Assuming that (I_i, M_i) is an ancestor attribute set, there does not exist another ancestor attribute set (I_j, M_j) that $(I_j, M_j) \leq (I_i, M_i)$, we say (I_i, M_i) is an least ancestor attribute set.*

Theorem 1. *An SSoD policy $e = ssod < P, U, k >$ is not enforceable if and only if the number of ancestor attribute sets for e is less than k.*

Proof. For the "if" part, we assume that if the condition in the theorem exists. Then one can construct a UCON_A state in which there are $k - 1$ users and each of the users is assigned one of the $k - 1$ ancestor attribute set (I, M) for e. Thus these $k - 1$ users together cover all m permissions, and result in an unsafe state. For the "only if" part, we show that if the condition in the theorem does not exist, then the SSoD configuration is enforceable. Consider that the number of ancestor attribute sets is k. we can declare every pair of (I, M) to be mutually exclusive, which forbids any user to cover any two of them, this makes $safe_e(\varepsilon)$ true. □

5 Enforcing SSoD Policies by SMEA Constraints

We now show that directly enforcing SSoD policies is intractable (coNP-complete).

Theorem 2. *The verification problem of $safe_e(\varepsilon)$ is coNP-complete.*

Proof. In ANSI RBAC model [14], a role is a collection of users and a collection of permissions, and the permission is a collection of object-right pairs. The $\mathrm{UCON_A}$ model can support RBAC in its authorization process, in $\mathrm{UCON_A}$, user-role assignment can be viewed as subject attributes and permission-role assignment as attributes of object and rights [5]. And let U in an SSoD policy $e = ssod < P, U, k >$ be the user set of all possible users in an RBAC state ε_0, then the verification problem of $safe_e(\varepsilon)$ is equivalent to the one in [12] for the theorem that checks whether an RBAC state is safe or not with respect to an SSoD policies, which is NP-complete. □

In RBAC, constraints such as mutually exclusive roles (SMER) are introduced to enforce SSoD policies [12]. We present a generalized form of the SMEA in this paper, which is directly motivated by SMER constraint.

Definition 4. *A statically mutually exclusive attribute (SMEA) constraint is expressed as*

$$smea < \{(I_1, M_1), \ldots, (I_m, M_m)\}, \{u_1, \ldots, u_n\}, k >$$

where each (I_i, M_i) is an attribute set and m and n are integers such that $2 \leq k \leq min(m, n)$.

Definition 5. *A* $\mathrm{UCON_A}$ *state ε is safe with respect to a SMEA constraint when*

$$\forall u_i \in \{u_1, \ldots, u_n\}(|(ATT_\varepsilon(u_i) \cap \{(I_1, M_1), \ldots, (I_m, M_m)\})| < k)$$

which means that no user from $\{u_1, \ldots, u_n\}$ is a member of k or more attribute sets in $\{(I_1, M_1), \ldots, (I_m, M_m)\}$, and we write it as $safe_c(\varepsilon)$. A $\mathrm{UCON_A}$ *state ε is safe with respect to a set C of SMEA constraints if it is safe with respect to every constraint in C, and we write it as $safe_C(\varepsilon)$.*

As each SMEA constraint restricts the attribute set memberships of a single user, it is efficient to check whether an $\mathrm{UCON_A}$ state satisfies a set of SMEA constraints, and thus provides a justification for using SMEA constraints to enforce SSoD policies.

5.1 Translating SSoD Policies to ASSoD Requirements

SMEA constraints are expressed in term of restrictions on attribute memberships, but SSoD policies are expressed in terms of restrictions on permissions. In order to generate SMER constraints for enforcing SSoD policies, the first step is to translate restrictions on attribute sets other than on permissions for SSoD policies. For each permission p_i in $\{p_1, \ldots, p_m\}$, there exists a (I_{p_i}, M_{p_i}) which is the threshold attribute set of p_i. In this way, we can define the attribute level SSoD requirement, and translate an SSoD policy to ASSoD requirements.

Definition 6. *An attribute level Static Separation-of-Duty (ASSoD) requirement is expressed as*

$$assod < \{(I_1, M_1), \ldots, (I_m, M_m)\}, \{u_1, \ldots, u_n\}, k >$$

where each (I_i, M_i) is an attribute set, m and n are integers such that $2 \leq n \leq m$.

Definition 7. *A* UCON$_\mathrm{A}$ *state ε is safe with respect to ASSoD requirement when*

$$\forall\{u_1^{'} \ldots u_{k-1}^{'}\} \subseteq \{u_1, \ldots, u_n\}(\bigcup_{i=1}^{k-1})ATT_\varepsilon(u_i^{'}) \nsupseteq \{(I_1, M_1), \ldots, (I_m, M_m)\}$$

It means that there should not exist a set of fewer than k users from $\{u_1, \ldots, u_n\}$ that together have memberships in all the m attribute sets in the requirement. A UCON$_\mathrm{A}$ *state ε is safe with respect to a set A of ASSoD requirements if it is safe with respect to every requirement in A, and we write it as $safe_A(\varepsilon)$.*

Let A denote the set of ASSoD requiremets derived from $< E, \varepsilon >$, if $< E, \varepsilon >$ is not enforceable, then it can not be translated to any ASSoD requirements, let $A = \emptyset$. Else if $< E, \varepsilon >$ is enforceable, then for each $e \in E$, and for each permission p_i in $\{p_1, \ldots, p_m\}$, there exist many attribute sets corresponding to it. Assume that each permission in $\{p_1, \ldots, p_m\}$ relates to the number of attribute sets is $\{k_1, \ldots, k_m\}$, then the total number of elements in $< A, \varepsilon >$ is $k_1 \times k_2 \times \ldots \times k_m$. Theorem 3 shows that for the ASSoD configuration $< A, \varepsilon >$ derived from an enforceable SSoD configuration $< E, \varepsilon >$ captures the same security requirement.

Theorem 3. *Given an SSoD configuration $< E, \varepsilon >$, and the ASSoD configuration $< R, \varepsilon >$ derived from $< E, \varepsilon >$, then $safe_A(\varepsilon) \Leftrightarrow safe_E(\varepsilon)$.*

Proof. Firstly, we show that if $safe_R(\varepsilon)$ is false, then $safe_E(epsilon)$ is also false. If $safe_R(\varepsilon)$ is false, then there exist $r = assod < \{(I_1, M_1), \ldots, (I_m, M_m)\}, U, n >$ and $k-1$ users that together cover all attribute sets in $\{(I_1, M_1), \ldots, (I_m, M_m)\}$. Given the way in which $ASSoD < R, \varepsilon >$ is derived from $< E, \varepsilon >$, there exists an SSoD policy in E such that the attribute set in R together have all the permissions in it, therefore $safe_E(\varepsilon)$ is also false. Secondly, we show that if $safe_E(\varepsilon)$ is false, then $safe_A(\varepsilon)$ is also false. If $safe_E(\varepsilon)$ is false, then there exist $e = ssod < \{p_1, \ldots, p_m\}, \{u_1, \ldots, u_n\}, k >$ and $k-1$ users together cover all permissions in $\{p_1, \ldots, p_m\}$. For each permission p_i in the permission set, there exists an attribute set (I_i, M_i) covering p_i, if it contains some sub attribute set, then we divide it, then there exists $r =< \{(I_1, M_1), \ldots, (I_m, M_m)\}, U, n >$ derived from e. Given the way in which $ASSoD < R, preA >$ is derived from $< E, \varepsilon >$, then $r \in R$, therefore $safe_A(\varepsilon)$ is also false. □

Theorem 4. *Given a* UCON$_\mathrm{A}$ *state ε, and a set A of ASSoD requirements, determine if $safe_A(\varepsilon)$ is coNP-complete.*

Proof. The proof is similar to the one in Theorem 2: let each attribute set in ASSoD requirement map to a permission. Then the ASSoD requirement is mapped to an SSoD policy. □

5.2 Generating SMEA Constraints to Enforce ASSoD Requirements

Definition 8. *Let C be a set of SMEA constraints, and R be a set of ASSoD requirement, C enforces R if and only if $safeC(\varepsilon) \Rightarrow safe_R(\varepsilon)$.*

Theorem 5. *The ASSoD requirement $a = assod < \{(I_1, M_1), \ldots, (I_m, M_m)\}, \{u_1, \ldots, u_n\}, k >$ can be enforced by the following SMEA constraint*

$$c = \bigcup_{i \neq j}^{i,j \in [1,m]} \{c = smea < \{(I_i, M_i), (I_j, M_j)\}, \{u_1, \ldots, u_n\}, 2 >\}$$

Proof. The ASSoD requirement means that k users are required to cover all m attribute sets. The constraint set C means that every two attribute sets in $\{(I_1, M_1), \ldots, (I_m, M_m)\}$ are mutually exclusive, then m users are needed to cover the m attribute sets, as $2 \leq n \leq m$, Thus $safe_{\{C\}}(\varepsilon)$ is true. □

Although the above SMEA constraints ($k = 2$) can enforce any ASSoD requirement, this may result in constraints that are more restrictive than necessary. Ideally, we want to generate SMEA constraints that can enforce the ASSoD requirement, and avoid generating constraints that are overly restrictive. For this, we prefer to use the less restrictive constraint set.

Definition 9. *Let C_1 and C_2 be two sets of SMEA constraints, C_1 is more restrictive than C_2 if $safe_{C_1}(\varepsilon) \Rightarrow safe_{C_2}(\varepsilon) \wedge safe_{C_2}(\varepsilon) \nRightarrow safe_{C_1}(\varepsilon)$, and we write it as $C_1 \succ_\varepsilon C_2$.*

We now give an algorithm to generate the relatively less SMEA constraints to enforce ASSoD requirement. Given an ASSoD requirement a, the first step is to compute the most restrictive SMEA constraint set C by enumerating all possible SMEA constraints (where m=k=2) ; the second step is to remove any constraint in C that the remainders can also enforce a; the third step is to weaken the more restrictive constraint in the set. This algorithm can be used by the step 1 and 2, which we only try to remove the constraints in C is efficient, the output C will be a relative less restrictive SMEA constraints to enforce a. Although by systematically enumerating all the cases in step 3 will generate the least restrictive SMEA constraints, the runtime will be expensive. We generally prefer to use step 1 and 2, and return the output C.

6 Conclusion

This paper presents two main contributions to the research of SoD policy in UCON$_A$: the specification and enforcement of SSoD in UCON. The specification is set-based and we show that it has simpler syntax than existing approaches. For the enforcement aspect, we have studied a number of problems related to generating SMEA constraints for enforcing SSoD policies in UCON$_A$ system. We show that directly enforcing SSoD policies in UCON$_A$ system is intractable (coNP-complete), study the problem how to verify whether a given SSoD configuration is enforceable, translate the SSoD policy to ASSoD requirement which be used as an intermediate step, and generate the least restrictive SMEA constraints from a set of ASSoD requirements. The results are fundamental to understanding the effectiveness of using constraints to enforce SSoD policies in UCON.

References

1. Clark, D., Wilson, D., Kuhn, D.R.: A Comparison of Commercial and Military Computer Security Policies. In: 8th IEEE Symposium on Security and Privacy, pp. 184–195. IEEE Press, Los Alamitos (1987)
2. Clark, D., Wilson, D., Kuhn, D.R.: Evolution of a Model for Computer Integrity. Technical Report, Invitational Workshop on Data Integrity, Section A2, pp. 1–3 (1989)
3. Ferraiolo, D.F., Kuhn, D.R., Chandramouli, R.: Role-Based Access Control. Artech House, 47–63 (April 2003)
4. Park, J., Sandhu, R.: The UCONABC Usage Control Model. ACM Transactions on Information and System Security 7(1), 128–174 (2004)
5. Zhang, X., Parisi-Presicce, F., Sandhu, R., Park, J.: Formal Model and Policy Specification of Usage Control. ACM Transactions on Information and Systems Security 8(4), 351–387 (2005)
6. Saltzer, J.H., Schroeder, M.D.: The Protection of Information in Computer Systems. Proceed Communications of the ACM 63(9), 1278–1308 (1975)
7. Brewer, D., Nash, M.: The Chinese Wall security policy. In: 10th IEEE Symposium on Security and Privacy, pp. 206–214. IEEE Press, California (1989)
8. Sandhu, R.: Transaction Control Expressions for Separation of Duties. In: 4th Annual Computer Security Applications Conference, pp. 282–286. IEEE Press, Orlando (1988)
9. Sandhu, R.: Separation of Duties in Computerized Information Systems. In: The IFIP WG11.3 Workshop on Database Security, pp. 18–21. IEEE Press, Halifax (1990)
10. Schaad, A., Lotz, V., Sohr, K.: A Model-checking Approach to Analyzing Organizational Controls in a Loan Origination Process. In: 11th ACM Symposium on Access Control Models and Technologies, pp. 139–149. ACM Press, California (2006)
11. Crampton, J.: Specifying and Enforcing Constraints in Role-based Access Control. In: 8th ACM Symposium on Access Control Models and Technologies, pp. 43–50. ACM Press, New York (2003)
12. Li, N., Tripunitara, M., Bizri, Z.: On Mutually Exclusive Roles and Separation-of-Duty. ACM Transactions on Information and System Security 10(2), 1–35 (2007)
13. Li, N., Mitchell, J.C., Winsborough, W.H.: Beyond Proof-of-Compliance: Security Analysis in Trust Management. Journal of the ACM 52(3), 474–514 (2005)
14. ANSI. American National Standard for Information Technology-Role Based Access Control. ANSI INCITS 359-2004 (2004)