Send Orders for Reprints to reprints@benthamscience.ae

Specification and Enforcement of the General User Authorization Query Problem in Role Based Access Control System

Xiaopu Ma[1,*], Yan Liu[1], Li Zhao[1], Yihua Lan[1] and Jianfeng Lu[2]

[1]*School of Computer and Information Technology, Nanyang Normal University, Nanyang, 473061, China*

[2]*School of Mathematics-Physical & Information Engineering, Zhejiang Normal University, Jinhua, 321004, China*

Abstract: The User Authorization Query (UAQ) problem in Role Based Access Control (RBAC) is assigning roles to users in an appropriate manner. That is, take as input a set of permissions that a user requests to have in a session, and determine whether there exists an optimum set of roles to active. However the existing definition of UAQ is inadequate, it only considers the number of permissions whereas the number of roles is also equally important, has been largely ignored. In addition, little attention has been paid to the complexity analysis of the UAQ problem with the consideration of the both permission and role numbers in the literature. In this paper, we give a general definition of UAQ with the name of GUAQ by introducing the consideration for the number of both permissions and roles, and then study the computational complexity of the GUAQ problem into three subcases. Furthermore, we propose an approach for finding a safe resolution for GUAQ, which employ the preprocessing and reduction to SAT solver that greatly reduce the running time.

Keywords: Computational complexity, RBAC, SAT, user authorization query.

1. INTRODUCTION

Role Based Access Control (RBAC) has recently received considerable attention and become the predominant approach for advanced access control [1]. RBAC has several beneficial features, such as policy neutrality, support for least privilege and efficient access control management, etc., which make RBAC better suited for handling access-control requirements of diverse organizations, and significantly simplify the management of users and permissions in computing systems [2]. The basic concept of RBAC is that permissions are associated with roles, and users are assigned to appropriate permissions via roles. The notion of roles adds a level of indirection to simplify the management of the many-to-many relation between users and permissions [3].

A fundamental problem in RBAC is assigning roles to users in an appropriate manner. In the real world, a user may be asked to perform a task only if he is qualified to do so, and the task responsibility is normally represented as a set of permissions in RBAC. In other words, if a role set R covers all the requested permissions, and all the roles in the R can be assigned to the user u, then u is qualified to do this task, thus this task can be accomplished. Zhang *et al.* refer it as the User Authorization Query (UAQ) problem [4], and the UAQ problem in RBAC systems has been widely studied for over a decade, several authors (e.g., Zhang [4], Wickramaarachchi [5], Armando [6], Chen [7], Mousavi [8], Lu [9] etc.) have focused on determining a set of roles to be activated in a single session in order to achieve some permissions while satisfying a collection of authorization constraints governing the activation of roles.

Surprisingly, most of the work focus on the objective of optimization on the number of permissions, such as how to minimize the number of extra permissions or missing permissions [10]. This is because missing some requested permissions may make the failure of the task, and too many extra permissions may bring intolerable risk to the system. However, the number of roles that are activated is another equally important optimization objective, has been largely ignored. To minimize the number of roles that are activated in a session is very important for the system, for example, minimal set of roles in sessions may be more efficient for the administration.

As discussed above, in this paper, we address the UAQ problem more comprehensively by defining the general UAQ (GUAQ) problem by considering the optimization objective for the number of both permissions and roles. That is, we want to not only minimize the number of extra permissions and missing permissions, but also the number of roles that are activated. Our contributions are summarized as follows:

1) We define a more general UAQ problem with the name of GUAQ, which takes the optimization on the number of both permissions and roles.

2) We study the computational complexity of the GUAQ problem into three subcases: any cover, safe cover and available cover, and show that all of them are intractable (NP-hard).

1874-4443/14 2014 Bentham Open

3) We propose an approach for finding a safe resolution for GUAQ, which employ preprocessing and reduction to SAT solver that greatly reduce the running time.

The rest of this paper is organized as follows. Section 2 gives the formal definition of GUAQ problem. Section 3 studies the computational complexity of the GUAQ problem into three subcases. Section 4 proposes an approach for GUAQ. We also discuss related work in Section 5, and conclude the paper in Section 6.

2. DEFINITION OF THE GENERAL USER AUTHORIZATION QUERY PROBLEM

Our definition of the GUAQ problem is inspired by that in [5], and generalizes it by introducing the consideration of the number of activated roles in a session. Our definition of the GUAQ problem taking the following three groups of information as input:

1) RBAC State Information: (*R*, *P*, *RP*) where *R*denotes the set of all roles in the system, *P* denotes the set of all permissions in the system, *RP* denotes the role-permission assignment relation and *Perm (r)* denotes all the permissions that be assigned to the role *r*, which includes the direct assignment and the permission inheritance.

2) The Permission Request Information: (P_{lo}, P_{up}, obj) where

(1)$P_{lo} \subseteq P_{up} \subseteq P$,$P_{lo}$ is the low bound for the set of requested permissions, which must be available. $P_{up}$is the upper bound for the set of requested permissions, where any permission not in P_{up} cannot be availed.

(2)$obj \in \{any, safe, available\}$means the optimization objective, where *any* means any permissions between P_{lo} and P_{up} is OK, and no requirement on the number of roles; *safe* means the number of permissions as close as to P_{lo} is better, and the smaller number of roles the better; *available* means the number of permissions as close as to P_{up} is better, and the smaller number of roles the better.

3) Security Constraints: *C* represents Dynamic Mutually Exclusive Role (DMER) constraints: $dmer\langle\{r_1, \cdots, r_n\}, k\rangle$ requires that no user can active k or more roles from $\{r_1, \cdots, r_n\}$.

The GUAQ problem outputs a role set $R_{sat} \subseteq R$ such that the following conditions hold:

1) $P_{lo} \subseteq Perm(R_{sat}) \subseteq P_{up}$;

2) All constraints in *C* are satisfied;

3) When *obj=any*, then R_{sat} need to satisfy the above two conditions, and for any R' that also satisfies the above two conditions, we have $|R'| \geq |R_{sat}|$;

4) When *obj=safe*, then for any R' that also satisfies the condition (1) and (2), we have $|Perm(R') \geq Perm(R_{sat})| \wedge |R'| \geq |R_{sat}|$;

5) When *obj=available*, then for any R' that also satisfies the condition (1) and (2), we have $|Perm(R')| < |Perm(R_{sat})|$ or $|Perm(R') = Perm(R_{sat})| \wedge |R'| \geq |R_{sat}|$.

3. COMPUTATIONAL COMPLEXITY OF GENERAL USER AUTHORIZATION QUERY PROBLEM

In the following, we study the computational complexity of the GUAQ problem into three subcases: any cover, safe cover and available cover. As prior work [8] points out, the minimization or maximization optimization objectives will impact the computational complexity of UAQ. However, they only show that the decision version of UAQ is in NP. To better understand how different optimization objectives may affect the complexity of GUAQ, we study the computational complexities of GUAQ into three subcases: any cover, safe cover, and available cover.

Definition 1. (*The any cover of GUAQ problem*) Given an RBAC state information(*R*, *P*, *RP*), permission request information (P_{lo}, P_{up}, obj), and a set of security constraints *C*, the any cover of GUAQ problem is a subcase of GUAQ where *obj=any*.

Theorem 1. The any cover of GUAQ problem is NP-hard.

Proof. Without loss of generality, we can simplify the any cover of GUAQ problem as follows: we first remove any role $r \in R$ from *R* if and only if $Perm(r) \nsubseteq P_{up}$; we second revise the DMER constraint $dmer\langle\{r_1, \cdots, r_n\}, k\rangle$ as follows: let $R' = \{r_1, \cdots, r_n\} \cap R$, and $k' = k - (n - |R'|)$. Without loss of generality, we assume that $R' = \{r_1, \cdots, r_m\}(m \leq n)$, and we get the renewed DMER constraint $dmer\langle\{r'_1, \cdots, r'_m\}, k'\rangle$, where $r'_i = r_i \cap P_{lo}$. It is obvious that the above transformation can be done in polynomial time.

We then show that the any cover of GUAQ problem without DMER constraints is NP-hard by reducing the NP-hard set cover optimization problem [11] to it. In the set cover optimization problem, the inputs are a finite set *U*, a family $S = \{S_1, \cdots, S_l\}$ of subsets of *U*. The goal is to find the smallest integer *m* for which there exists an exact cover of *U* of cardinality *m*. The reduction is as follows. Given *S* and *U*, we simply set $\{r'_1, \cdots, r'_m\}$=*U* and P_{lo}=*S*. Clearly, a solution R_{sat} to the any cover of GUAQ problem provides a solution to the set cover optimization problem.

It is obvious that the any cover of GUAQ problem is at least as hard as the any cover of GUAQ problem without DMER constraints, because the latter problem may return an answer R_{sat} for the any cover of GUAQ problem if and only if all constraints in C are satisfied. Determine whether a given DMER constraint $dmer\langle\{r'_1, \cdots, r'_m\}, k'\rangle$ is satisfied can be done in polynomial time: one first counts how many roles in R_{sat} also appear in $\{r'_1, \cdots, r'_m\}$, and finally copares this number with k'.

As shown above, the any cover of GUAQ problem is NP-hard.

Definition 2. (*The safe cover of GUAQ problem*) Given an RBAC state information (*R*, *P*, *RP*), permission request information (P_{lo}, P_{up}, obj), and a set of security constraints *C*, the safe cover of GUAQ problem is a subcase of GUAQ where *obj=safe*.

Theorem 2. The safe cover of GUAQ problem is NP-hard.

Proof. We first remove any role $r \in R$ from R if and only if $Perm(r) \nsubseteq P_{up} \vee Perm(r) \cap P_{lo} = \emptyset$. We then configure a special case of safe cover of the GUAQ problem as follows: given R,P_{lo}, the goal is to find a role set R_{sat} such that $Perm(R_{sat}) \supseteq P_{lo}$ and $Perm(R_{sat})$ is minimized. This special case does not consider the DMER constraints, and also does not consider the optimization of role numbers.

In order to show the safe cover of GUAQ problem is NP-hard, we reduce a special case of it to the NP-hard container optimization problem [12] to the special case. In the container optimization problem, the inputs are a finite set X, a family $C = \{C_1, \cdots, C_l\}$ of subsets of X, and $V \subseteq X$, the goal is to determine whether there exists a container T of V such that $|T|$ is minimized, where T denotes a special any cover of T in C where $P_{lo} = P_{up}$. Let (P_{up}, R,P_{lo}) be an instance of the special case of the safe cover of GUAQ problem, we transform it into an instance (X, C, V) of the container decision problem as follows: let P_{up}=X, R=C, and P_{lo}=V. We query an oracle to obtain a solution R_{sat} of P_{lo} such that $Perm(R_{sat}) \supseteq P_{lo}$ and $Perm(R_{sat})$ is minimized, then we simply compute $\bigcup_{r \in R_{sat}} Perm(r)$, which is a container T of V such that $|T|$ is minimized.

Definition 3. (*The available cover of GUAQ problem*) Given an RBAC state information (R, P, RP), permission request information (P_{lo}, P_{up}, obj), and a set of security constraints C, the available cover of GUAQ problem is a subcase of GUAQ where *obj=available*.

Theorem 3. The available cover of GUAQ problem is NP-hard.

Proof. We first remove any role $r \in R$ from R if and only if $Perm(r) \nsubseteq P_{up}$. We then let $P_{req} = Perm(R)$. Of course this can be done in polynomial time. A special case of the available cover of GUAQ problem is that en R, P_{req} ,the goal is to find a role set R_{sat} such that $Perm(R_{sat}) = P_{req}$ and $|R_{sat}|$ is minimized. This special case does not consider the DMER constraints. In order to show the available cover of GUAQ problem is NP-hard, we reduce the NP-hard set cover optimization problem to it. Similar to the proof in Theorem 1, we simply set R=S and P_{req}=U. Clearly, a solution R_{sat} to the special case of the available cover of GUAQ problem provides a solution to the set cover optimization problem.

4. AN APPROACH FOR THE GENERAL USER AUTHORIZATION QUERY PROBLEM

In the last section, we have studied the computational complexities of GUAQ into three subcases: any cover, safe cover, and available cover, and show that all of them are NP-hard, which means there exist difficult problem instances that take exponential time in the worst case. Many instances that will be encountered in practice may still be efficiently solvable. In the rest of this section, we describe an approach for the safe cover of GUAQ problem (we write it as S-GUAQ for short), which employs the following two techniques that greatly reduce the running time. Of course, this approach can be extended to support the any cover and safe cover of GUAQ problems.

1) We employ a preprocessing technique that aims at reducing the size of the S-GUAQ problem.

2) We translate S-GUAQ problem into an SAT instances, which enable us to benefit from the extensive research on SAT and to use existing SAT solver to reduce the running time.

4.1. Preprocessing for S-GUAQ

Given an RBAC state information (R, P, RP), permission request information(P_{lo}, P_{up}, obj), and a set of security constraints C. We try to reduce the size of S-GUAQ, we first reduce the number of roles in the system as follows:

1) We first reduce the number of roles in the systems as follows: For every role $r \in R$, we remove it from R if and only if $Perm(r) \nsubseteq P_{up}$ or $Perm(r) \cap P_{lo} = \emptyset$.

2) Secondly, we try to reduce the number of security constraints as follows: For every DMER constraint $dmer\langle\{r_1, \cdots, r_n\}, k\rangle$, if $\{r_1, \cdots, r_n\} \cap R < k$, that is, $dmer\langle\{r_1, \cdots, r_n\}, k\rangle$ is always satisfied, thus we can remove it from C.

4.2. Reduction S-GUAQ to SAT

In this section, we show how the S-GUAQ problem can be efficiently reduced to an SAT problem, which enables us to benefit from the extensive research on SAT and to use existing SAT solvers, such as SAT4j [13].

Given:

1) R: the set of all roles in the RBAC system that are available to the user;

2) P: the set of all possible permissions in the RBAC system;

3) RP: the set of role-permission assignments $\{(r_i, p_j)|p_j \in P \wedge p_j \in Perm(r_i)$ or $\exists r_j \in R, r_i \geq r_j \wedge p_j \in Perm(r_j)\}$;

4) P_{lo}: the low bound for the set of requested permissions;

5) P_{up}: the upper bound for the set of requested permissions;

6) C: the set of DMER and cardinality constraints of the form $\langle\{r_1, \cdots, r_n\}, k\rangle$ (The cardinality constraint $cd\langle r, t\rangle$ can be regardes as a special case of DMER constraint $dmer\langle r, t\rangle$)

Formalization:

1) For each $r_i \in R$, let r_i^v denote a SAT variable which is true if and only if r_j is activated.

2) For each $p_j \in P$, let p_j^v denote a SAT variable which is true if and only if p_i is activated.

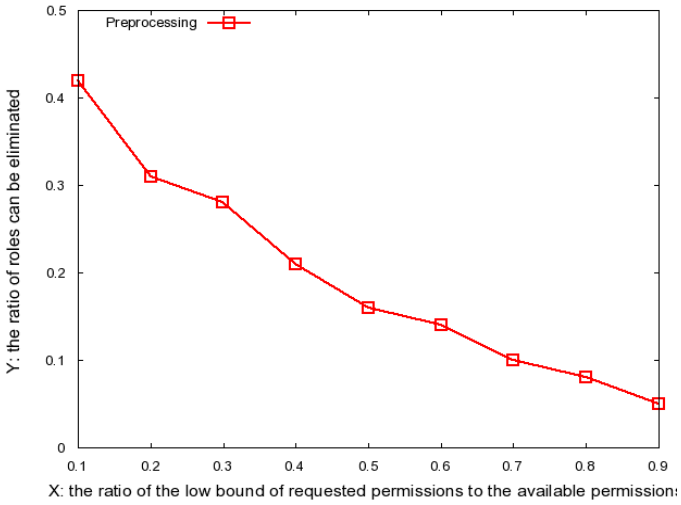


Fig. (1). Effective of preprocessing for different ratios of P_{lo} to P.

Processing Steps:

1) S=∅;

2) For each $p_i \in P_{lo}$, mark v_{p_i}as a non-relaxable clause and add it to S;

3) For each role $r_k \in R$, construct an equivalence propositional clause of the form $v_{r_k} \rightarrow (p_i \wedge \cdots \wedge p_j)$, mark it as non-relaxable and add it to S;

4) For each permission $p_l \in P$, construct a clause of the form $v_{p_l} \rightarrow (r_i \vee \cdots \vee r_j)$, mark it as non-relaxable and add it to S;

5) For each constraint $\langle \{r_1, \cdots, r_n\}, k\rangle \in C$, we use Pseudo-Boolean (PB) constraints to describe the DMER and cardinality constraints. In PB constraints, all variables take values of either 0 (false) or 1 (true). Constraints are linear inequalities with integer coefficients. For each DMER constraint $dmer\langle \{r_1, \cdots, r_n\}, k\rangle$, we specify a PB constraint $\sum_{r_i \in \{r_1, \cdots, r_n\}} v_{r_i} < k$.

6) To favor minimize the number of permissions: apply the previous SAT formulation, but for each$p_j \in P_{up} \backslash P_{lo}$, mark $\neg v_{p_j}$ as a relaxable clause and add it to S;

7) To favor minimize the number of roles: apply the previous SAT formulation, but for each $r_i \in R \wedge Perm(r) \subseteq P_{up}$, mark $\neg v_{r_i}$ as a non-relaxable clause and add it to S;

8) Solve S uses SAT4j, if it returns SATISFIABLE, then active each role $r_m \in R$ such that v_{r_m} is set to true.

4.3. Experimental Evaluations

We design experimental evaluations to show the effectiveness of the proposed approach. Experiments are carried out on a desktop PC with an Intel Core i7-2600 running at 3.4GHz, and with DDR3 4GB 1333MHz, running Microsoft Windows 7 Home Basic. The configurations that we use to generate testing instances are as follows:

1) The ratio of roles to permissions is 1:5;

2) The ratio of the low bound of requested permissions P_{lo} to all available permissions in the system is from 1:10 to 9:10;

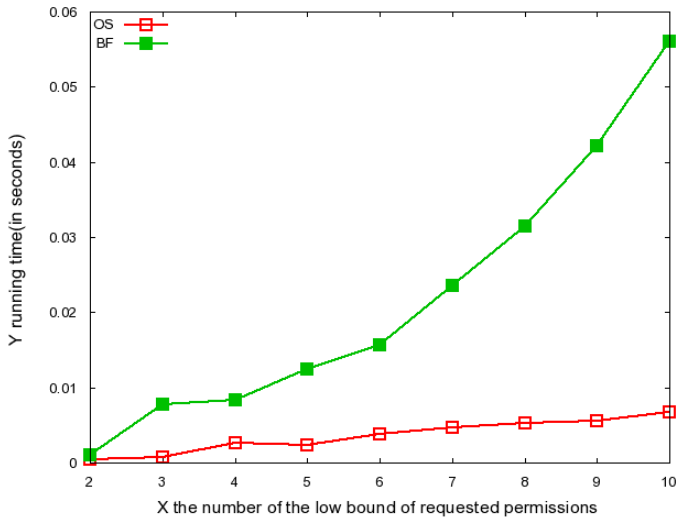
3) Let the upper bound of requested permissions P_{up} equal to the permissions in the system;

4) For each instance, 10 randomly generated test cases were run and the results were used to generate the graphs.

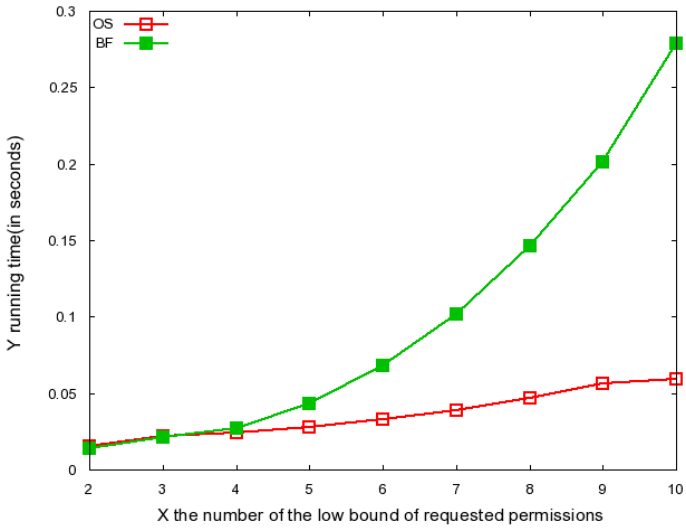
Preprocessing is effective: Fig. (**1**) shows the effective of preprocessing for different ratios of low bound of requested permissions to permissions. When the ratio is small, that is, the set of low bound of requested permissions is only a small part of the available permissions in the system, the preprocessing perfects very well. However, as the ratio increases, preprocessing is less effective, this is because that less roles can be removed from the consideration. For example, when the ratio of low bound of requested permissions to permissions is 10%, more than 42% roles can be reduced by preprocessing, which can decrease the number of roles that should be considered in the problem of S-GUAQ. And when the ratio of low bound of requested permissions to permissions is 90%, less than 5% roles can be removed from consideration.

Compare the Optimization SAT based (OS) approach with the Brute-Force (BF) approach: In order to understand the effectiveness of our approach, we have implemented two approaches: one is our OS approach, the other is the BF approach from [5] on four different role permission assignments.

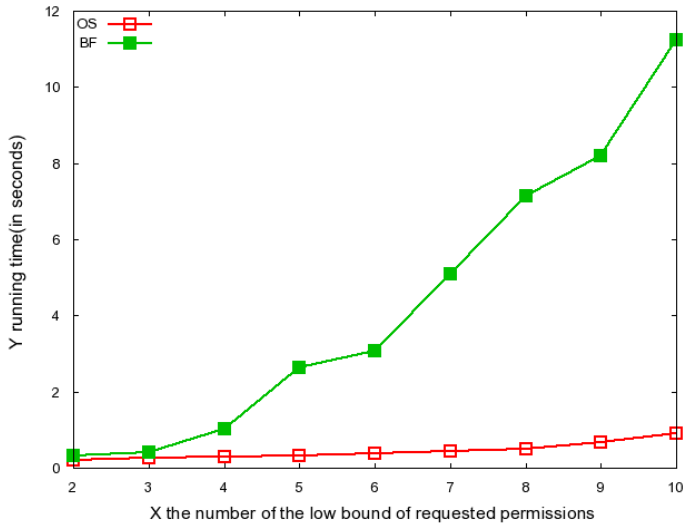
Fig. (**2a-d**) shows the result of running the experiments for the two approaches. When the number of low bound of requested permissions requested is small, the two approaches perform produce comparable results. As the number of low bound of requested permissions increase, the overall trend in time taken increases exponentially makes the BF approach impractical for implementation in dynamic systems. On the other hand, OS approach takes a few seconds, even for a



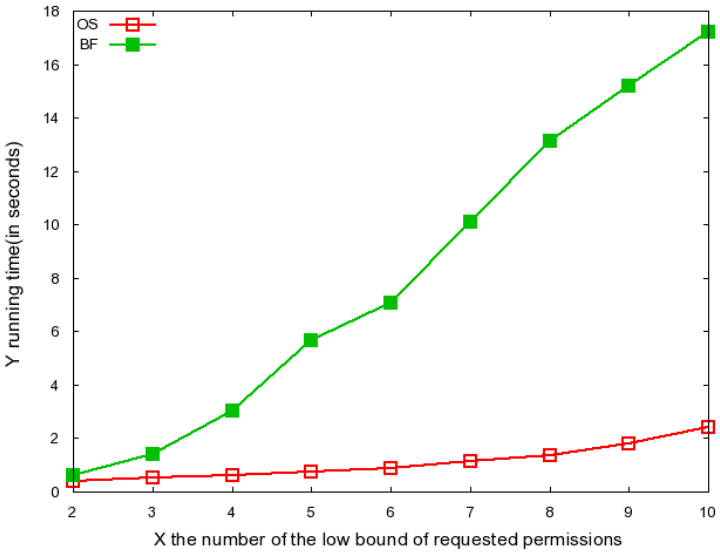
(a) 10 roles and 50 permissions



(b) 20 roles and 100 permissions



(c) 40 roles and 200 permissions



(d) 60 roles and 300 permissions

Fig. (2). Running time for the Optimization SAT (OS) approach and Brute-Force (BF) approach.

larger number of roles, permissions and permissions requested.

5. RELATED WORK

This paper has studied the GUAQ problem. The similar problem has been proposed by Zhang and Joshi with the name of the User Authorization Query (UAQ) problem [4], that is, to determine the set of roles to be activated in a single session for a particular set of permissions requested by the user. This set of roles must satisfy security constraints that prevent certain combinations of roles to be activated in one session, and should follow the least privilege principle. They proposed a two-step algorithm for the UAQ problem. However, this algorithm has some false negatives, such as falsely rejecting some legal success. Wickramaarachchi *et al.* [5] provided a more general definition of UAQ problem where the permission grant includes both a lower bound and an upper bound. However, they did not consider the number of roles as an optimization objective for the UAQ problem, which is also equally important, has been ignored. They provided two approaches to UAQ. In the first one, they employed the Davis-Putnam-Logemann-Loveland (DPLL) algorithm for solving the CNF-SAT problem [14]. In the second approach, they reduced the UAQ problem to the MAXSAT problem, and used optimized off-the-shelf SAT solvers such as zChaff.

In order to scale to larger RBAC policies, Armando et al [6] described an SAT-based technique to solve the UAQ problem which overcomes this limitation. They carefully tune the reduction to the SAT problem so that most of the clauses need not to be generated at run-time but only in a preprocessing step. Another similar problem is proposed by Du *et al.* with the name of inter-domain role mapping (IDRM) problem [10], which relates to determining whether a user's request for activating a set of roles can be granted or not. Chen [12] constructed a series of role-based models and used these models to investigate least privilege and the IDRM problem in the presence of multiple role hierarchies and temporal constraints.

Although the similar problems (e.g., UAQ, IDRM) are well-studied, the existing works do not always pose the most appropriate problem, as UAQ considered DMER constraints rather than DSoD policies that affect the solution of UAQ. Li *et al.* [15] considered that the distinction between DSoD policies as objectives and DMER constraints as a mechanism is not clearly will raise the security risks. One danger is that it is unclear whether the higher-level objectives are met by the constraints or not, this is because the DMER constraints may be specified without a clear specification of what objectives they intend to meet; Another danger is that even though when DMER constraints are specified, and there exists a clear understanding of what DSoD policies are desired, when the assignment of permissions to roles changes, the DMER constraints may no longer be adequate for enforcing the desired DSoD policies [16]. Most of all, existing work does not consider the number of roles as an optimization objective of the UAQ problem. As pointed out in Section 1, the optimization on the number of roles is another important optimization objective, it is help to efficiently management the systems. However, it has been largely ignored. Moreover, little attention pays for the complexity analysis of the UAQ problem with optimization of role number. Previous work [4, 5] has shown that the UAQ problem in general is NP-hard.

As prior work [8] points out, the minimization or maximization optimization objectives will impact the computational complexity of UAQ. However, they only show that the decision version of UAQ is in NP. To better understand how different optimization objectives may affect the complexity of GUAQ, we study the computational complexities of GUAQ into three subcases: any cover, safe cover, and available cover. Chen *et al.* [12] assert that allowing both options of minimization and maximization is no more difficult than allowing minimization only. However, Mousavi *et al.* [8] show that this assertion is not true. For example, with the introduction of constraints, if $P \neq NP$, the sub-case of UAQ

that maximizes the number of extra permissions is no longer tractable. Therefore, to better understand how different optimization objectives may affect the complexity of GUAQ is necessary. Our work is an important supplement to the research of UAQ problem in RBAC systems.

CONCLUSION

In this paper, we give a more general definition of the UAQ problem with the name of GUAQ, which takes the optimization on the number of both roles and permissions. We study the computational complexity of the GUAQ problem into three subcases: any cover, safe cover and available cover, and show that all of them are intractable. We also propose an approach for S-GUAQ by employing preprocessing and reduction to SAT techniques that greatly reduce the running time.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interests regarding the publication of this article.

ACKNOWLEDGEMENTS

This work is supported by the Joint Fund for National Natural Science Foundation of China and Henan Province for Fostering Talents under grants U1304619, National Natural Science Foundation of China under grants 61402418, MOE (Ministry of Education in China) Project of Humanity and Social Science under Grant 12YJCZH142 and Innovation Fund of Nanyang Normal University under grants ZX2013013. We thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] ANSI, American national standard for information technology-role base access control (ANSI INCITS 359-2004), 2004.

[2] X. Ma, R. Li, Z. Lu, J. Lu and M. Dong, "Specifying and enforcing the principle of least privilege in role-based access control", *Concurrency and Computation: Practice & Experience,* vol. 23, no. 12, pp. 1313-1331, 2011.

[3] R. Sandhu, V. Bhamidipani, and Q. Munawer, "The arbac97 model for role-based administration of roles", *ACM Transactions on Information and System Security* (TISSEC), vol. 2, no. 1, pp. 105-135, 1999.

[4] Y. Zhang and J. B. D. Uaq "A framework for user authorization query processing in rbac extended with hybrid hierarchy and constraints", *Proceedings of the 13th ACM symposium on Access control models and technologies* (SACMAT), New York, USA, pp. 83-92, 2008.

[5] T. G. Wickramaarachchi, H. Q. Wahbeh, and N. Li, "An Efficient Framework for User Authorization Queries in RBAC System", *Proceedings of the 14th ACM symposium on Access control models and technologies* (SACMAT), (Stresa, Italy), pp. 23-32, 2009.

[6] A. Armando, S. Ranise, F, Turkmen and B. Crispo, "Efficient Runtime Solving of RBAC User Authorization Queries: Pushing the Envelope", *Proceeding of the ACM Conference on Data and Application Security and Privacy*, (San Antonio, Texas, USA, pp. 241-248), 2012.

[7] L. Chen and J. Crampton, "Inter-domain role mapping and least privilege", In: *Proceeding of the 12th ACM Symposium on Access Control Models and Technologies* (SACMAT), (Sophia Antipolis, France), pp. 157-162, June 2007.

[8] N. Mousavi and M. V. Tripunitara, "Mitigating the Intractability of the User Authorization Query Problem in Role-Based Access Control (RBAC)", *Proceedings of the 6th International Conference on Network and System Security*, (Fujian, China), pp. 516-529, 2012.

[9] J. Lu, J. Han, W. Chen and J. Hu, "Safety and availability checking for user authorization queries in RBAC", *International Journal of Computational Intelligence Systems,* vol. 5, no. 5, pp. 860-867, 2012.

[10] S. Du, and J.B.D. Joshi, "Supporting authorization query and inter-domain role mapping in presence of hybrid role hierarchy", *Proceeding of the 11th ACM Symposium on Access Control Models and Technologies* (SACMAT, pp. 228-236), 2006.

[11] M.R. Garey, and D.J. Johnson, "*Computers and Intractability: A Guide to the Theory of NP-Completeness*", Freeman, San Francisco, CA, 1979.

[12] L. Chen and J. Crampton, "Set Cover Problems In Role-Based Access Control", *Proceeding of the 14th European Symposium on Research in Computer Security*, (Springer-Verlag Berlin, Heidelberg, pp. 689-704), 2009.

[13] D. Le Berre (project leader), SAT4J: A satisfiability library for Java, URL http://www.sat4j.org/, January 2006.

[14] C. Sinz, "Visualizing sat instances and runs of the dpll algorithm" *Journal of Automation Reasoning*, vol. 39, no. 2, pp. 219-243, 2007.

[15] N. Li, M. Tripunitara, and Z. Bizri, "On mutually exclusive roles and separation-of-duty", *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 2, pp. 1-35, 2007.

[16] J. Lu, R. Li, J. Hu, and D. Xu, "Static enforcement of static separation-of-duty policies in usage control authorization models", *IEICE Transactions on Communications,* vol. E95-B, no. 5, pp. 1508-1518, 2012.

Received: November 20, 2014 Revised: January 08, 2015 Accepted: January 19, 2015