A Conflict-Related Rules Detection Tool for Access Control Policy

Xiaoyan Liang, Liangshuang Lv, Chunhe Xia, Yang Luo, and Yazhuo Li

Key Laboratory of Beijing Network Technology
School of Computer Science and Engineering, Beihang University, Beijing, China
lxy@cse.buaa.edu.cn, {lls,xch}@buaa.edu.cn,
veotax@sae.buaa.edu.cn, lyzalexandra@126.com

Abstract. Conflict detection is an important issue of the Access Control Policy. Most conflict detection tools mainly focus on the two rules that have contrary actions, but there are also other rules which are necessary to the conflict situation, which is not considered in these tools. This paper defines all these rules related to the conflict situation as the concept "conflict-related rules", and gives a conflict-related rules detection tool for Access Control Policy which can report the conflict situation more comprehensively. By giving the semantics model of the access control policy and the definition of conflict, we prove the necessary and sufficient condition of conflict, and then give the concept of "conflict-related rules" and deduce its extension. We implement conflict-related rules detection tool based on the description logic, and the experiment results validate the tool's correctness and effectiveness. The results of the correctness experiment showed that instead of detecting the two rules with opposite actions only, it detected all the conflict-related rules for access control policy; the results of the effectiveness experiment showed that our tool's response performance is better than VPN based tools.

Keywords: Access control policy, conflict detect, conflict-related rules, description logic.

1 Introduction

Policy based access control is an important part of network information security [1,12]. An access control policy is a list of access control rules. The rules may conflict when they declared opposite access control behaviors. Conflicts in a policy can cause hole in security or block legal access. Conflict detection is an important issue for access control policy. Tools for conflict detection give many conflict detection algorithms under various scenarios, and they can report the two rules which have opposite actions.

However, reporting the two rules only can't help security administrator fully understand the situation of conflict. Take the AC (access control) policy of an enterprise information management system as an example, the enterprise has two kinds of users: server and marketer, and three kinds of accessible resources: user-information, contact-information and privacy-information.

J. Su et al. (Eds.): ICoC 2013, CCIS 401, pp. 158–169, 2013.

Security administrator configures following access control rules: r0: server inherit marketer; r1: permit marketer read user-information; r2: permit marketer delete contact-information; r3 deny server read privacy-information; r4 user-information contain contact-information; r5 user-information contain privacy-information.

Rule 1 and rule 3 have semantic conflict: rule 1 indicates that server inherits the authority of marketer and has the permission to access user-information which includes privacy-information; while rule 3 forbids server to access privacy-information.

Security administrator expects to understand not only rule 1 and rule 3 which have opposite actions, but also all rules related to the situation of "conflict". Rule 0 and rule 5 from the example above are also the causes of "conflict" and conflict cannot happen without these rules. We name these rules, which can cause conflict indirectly, as "Conflict-related Rules".

The contributions of this paper are:

1) We abstract all the rules in the conflict situation, not only the two rules that have contrary actions, as the concept of the "conflict-related rules", and we deduce this concept's extension.

2) Based on description logic, we implement the conflict detection tool based to detect the "conflict-related rules".

This paper is organized as follows: Section 2 describes the related work and discussion. Based on the semantic formal representation of access control policy, we give the necessary and sufficient condition of "conflict" and deduce the extension of "conflict-related rules", which makes the range of "conflict-related rules" explicit in section 3. Based on description logic, section 4 Figure and Table shows the implement of the conflict-related rules detection tool. Experiments in section 5 validate the correctness and effectiveness of the tool and Section 6 presents the conclusion.

2 Related Work

There are several researches on conflict detection of AC policy at present. Lupu and Sloman proposed a conflict detection tool focusing on authorization policy and obligation policy [3], they suggested that the rule of conflict is two rules which have opposite actions. He lili presented a conflict detection tool which is based upon OWL and RBAC negative authorization [4], which just concerns rules have opposite actions. Jianfeng Lu etc. studied two kinds of conflict of access control policy in the multi-domain environment [5]. Chang-Joo Moon did research on conflict among permission assignment constraints (PAC) in RBAC [6]. Basit Shafiq studied conflict between RBAC policies of each domain in multi-domain environment for collaborative work of multiple organizations [7]. Feng Huang etc. presented a description logic based conflict detection tool for access control policy. After the management of XACML access control policy, reference [8,9] converts the detection problem of XACML policy conflict into the consistency of knowledge base for description logic. Apurva Mohan etc. proposed a terminology based conflict detection method of authorization policy, which uses ontology reasoning to detect the conflict, and the detected "conflict" is defined by

existing concept [10]. Mansor et al. give the dynamic conflict detection algorithm for policy-based management [11].

Therefore, we discover that researches on conflict detection tool for access control policy at present are mainly focusing on two rules which have opposite actions, but ignoring other "conflict-related rules". It is necessary to research the extension of "conflict-related rules" and the detection tools.

3 Semantic Model for AC Policy

The grammar representation of access control policy is given first, and then we analyze its semantics, giving its semantic formal representation. For convenience of expression, we use "policy" instead of "access control policy" and "rule" instead of "access control rule".

3.1 Grammatical Formal Model for Access Control Policy

Definition 1: Access Control Policy (grammatical definition): Grammatically, policy is the set of rule statements, and a rule statement comprised of components complying with grammar rules.

Grammatically, the formal representation of access control policy is as follows:

$$\begin{cases} \text{P-STATE} = \left\{ r-state \mid r-state = \begin{cases} \langle sub-state, sub-state \rangle, \\ \langle obj-state, obj-state \rangle, \\ \left\langle \begin{matrix} sub-state, obj-state, act-state \\ , perm-state \end{matrix} \right\rangle \end{cases} \right\}; \\ sub-state, obj-state, act-state, perm-state \in \{ exp \mid grammer(exp) \}; \end{cases} \tag{1}$$

Where, P-STATE is the set of rule clarifications, namely policy; $r-state$ is a rule clarification in policy; $sub-state$ is the subject expression; $obj-state$ is the object expression; $act-state$ is the action expression; $perm-state$ is the permission expression. $grammer(exp)$ is a predicate, indicating that the expression correspond with the specification of grammar.

3.2 Semantic Formal Model for Access Control Policy

According to grammar of access control policy presented in part A, we first analyze the implication of expressions, and then analyze the semantics of rule statement of the three types, which will finally deduce the semantics of policy.

Definition 2: Semantics Expressed by "Subject Expression" and "Object Expression": Semantics are specific entities. Semantics of subject expression is users or

characters affected by policy. Semantics of object expression is resources protected by policy. The formal representation of set is as follows:

$$\begin{cases} \text{SUBJECT}=\{sub|\text{U}(sub)\}; \\ \text{OBJECT}=\{source|\text{R}(source)\}; \end{cases} \tag{2}$$

Where, SUBJECT is the set, representing the set of semantics for subject expression $sub-state$; $\text{U}(sub)$ is the predicate, representing sub is a user or character; $\text{R}(source)$ is the predicate, representing $source$ is the resource protected by policy.

Definition 3: Semantics of Statements Comprised of "Subject Expression" and "Subject Expression": Rules of this type are inheritance relationship in essence, so rule semantics are expressed by using the relationship between semantics of subject expression. The representation is as follows:

$$\xi_{inherit} \in \text{SUBJECT} \times \text{SUBJECT}; \tag{3}$$

Where, $\xi_{inherit}$ represents the semantics of rule on subject. The definition of SUBJECT is as formula (2).

Definition 4: Semantics of Statements Comprised of "Object Expression" and "Object Expression": its semantics represents inclusion relation between objects (namely, protected resources), which is expressed as:

$$\xi_{contain} \in \text{OBJECT} \times \text{OBJECT}; \tag{4}$$

Where, $\xi_{contain}$ represents the semantics of object relation rule. The definition of OBJECT is as formula (2).

Definition 5: Semantics of Statements Comprised of "Object Expression", "Object Expression", "Action Expression" and "Permission Expression":

Semantics of subject expression is subject (a user or a character); Semantics of object expression is object (protected resources); Semantics of action expression enable actions that subject can do to object (namely, operations like read, write, etc.), and it has different extensions according to different systems. Semantics of permission expression is "permit" and "deny". Therefore, semantics of this kind of rule is the actions taken by subject on object to "permit" or "deny" some kind of operation. So this kind of operation can be expressed as the relationship. The direction of the relationship represents "permit" or "deny". The "permit" is expressed as a directed two-tuple of "from subject to object" and the "deny" is expressed as a directed two-tuple of "from object to subject". Various "actions" are usually declared in access control. Each action and its "permission" will be represented by a directed relationship. There are k kinds of actions. Its formal representation is as follows:

$$\xi_{\text{ACTION}k} \subseteq \text{SUBJECT} \times \text{OBJECT} \cup \text{OBJECT} \times \text{SUBJECT}; k=1,2,.....,n; n \in N; \tag{5}$$

Definition 6：Access Control Policy (Semantic Definition):
Semantics of access control policy consists of semantics of access control rules. So the semantic formal representation of access control policy is as follows:

$$\begin{cases} \text{POLICY} = \left(\begin{matrix} \xi_{\text{INHERIT}}, \xi_{\text{CONTAIN}}, \xi_{\text{ACTION1}}, \\ \xi_{\text{ACTION2}}, \cdots, \xi_{\text{ACTION}n} \end{matrix} \right); \\ \xi_{\text{INHERIT}} \subseteq \text{SUBJECT} \times \text{SUBJECT}; \\ \xi_{\text{CONTAIN}} \subseteq \text{OBJECT} \times \text{OBJECT}; \\ \xi_{\text{ACTION}k} \subseteq \text{SUBJECT} \times \text{OBJECT} \cup \text{OBJECT} \times \text{SUBJECT}; \\ 1 \le k \le n; n \ge 1; \end{cases} \tag{6}$$

Where, POLICY represents the semantic of access control policy, which includes the inheritance relationship of subject ξ_{INHERIT}, the relationship between objects ξ_{CONTAIN} and the relationship between subject and object $\xi_{\text{ACTION1}}, \xi_{\text{ACTION2}}, \ldots, \xi_{\text{ACTION}n}$.

The semantics between rules of access control policy is implicit, since there are inheritance relationship and inclusion relationship between the rules. The implicit semantics of access control policy between subjects having inheritance relationship is expressed as axiom 1, 2 and 3.

Axiom 1： Relationship of ξ_{INHERIT}, ξ_{CONTAIN} is reflexive and transitive.

Axiom 2：Semantics implied by the inheritance relationship between subjects is:

For any $subject_i$, $subject_j$, $object_k$:

（1）if $\langle subject_i, subject_j \rangle \in \xi_{\text{INHERIT}}$ and $\langle subject_j, object_k \rangle \in \xi_{\text{ACTION}k}$, then: $\langle subject_i, object_k \rangle \in \xi_{\text{ACTION}k}$

（2）if $\langle subject_i, subject_j \rangle \in \xi_{\text{INHERIT}}$ and $\langle object_k, subject_j \rangle \in \xi_{\text{ACTION}k}$, then: $\langle object_k, subject_i \rangle \in \xi_{\text{ACTION}k}$

Axiom 3：Semantics implied by the inclusion relationship between objects is:

For any $subject_l$, $object_m$, $object_n$:

if $\langle object_m, object_n \rangle \in \xi_{\text{CONTAIN}}$ and $\langle subject_l, object_m \rangle \in \xi_{\text{ACTION}k}$, then: $\langle subject_l, object_n \rangle \in \xi_{\text{ACTION}k}$

（2）if $\langle object_m, object_n \rangle \in \xi_{\text{CONTAIN}}$ and $\langle object_m, subject_l \rangle \in \xi_{\text{ACTION}k}$, then: $\langle object_n, subject_l \rangle \in \xi_{\text{ACTION}k}$

3.3 Conflict-Related Rules

Conflict.

Definition 7 Conflict in Access Control Policy: The conflict discussed in AC policy is that two rules with the same subjects and objects but have opposite actions.

From the characteristic "with the same subjects and objects have opposite actions" in Definition 7, we know that two access control rules of conflict are all statements consisting of "subject expression", "object expression", "action expression" and "permission expression", and in the semantics they expressed, the actions are from the same type but opposite.

Theorem 1 The Necessary and Sufficient Condition of Conflict in Access Control Policy:

For $\xi_1 = \langle subj_1, obje_1 \rangle$, $\xi_2 = \langle obje_2, subj_2 \rangle$, the necessary and sufficient condition for conflict of ξ_1 and ξ_2 is:

(1) condition 1 $\exists x \in$ SUBJECT that:

$$\langle x, subj_1 \rangle \in \left(\xi_{\text{INHERIT}}\right)^m \wedge \langle x, subj_2 \rangle \in \left(\xi_{\text{INHERIT}}\right)^n ; m, n \geq 0; m, n \in N;$$

(2) condition 2 $\exists y \in$ OBJECT that:

$$\langle obje_1, y \rangle \in \left(\xi_{\text{CONTAIN}}\right)^j \wedge \langle obje_2, y \rangle \in \left(\xi_{\text{CONTAIN}}\right)^k ; j, k \geq 0; j, k \in N;$$

Proof:

- the Proof of Sufficient Condition

$$\frac{\langle x, subj_1 \rangle \in \left(\xi_{\text{INHERIT}}\right)^m, \text{axiom 1}}{\langle x, subj_1 \rangle \in \xi'_{\text{INHERIT}}}, \frac{\langle x, subj_1 \rangle \in \xi'_{\text{INHERIT}},\ \xi_1 = \langle subj_1, obje_1 \rangle \ \text{axiom 2}}{\langle x, obje_1 \rangle}$$

$$\frac{\langle x, obje_1 \rangle, \langle obje_1, y \rangle \in \left(\xi_{\text{CONTAIN}}\right)^k, \text{axiom 3}}{\langle x, y \rangle}$$

Thus, $\langle x, y \rangle$ was deduced .

$$\frac{\langle x, subj_2 \rangle \in \left(\xi_{\text{INHERIT}}\right)^m, \text{axiom 1}}{\langle x, subj_2 \rangle \in \xi'_{\text{INHERIT}}}, \frac{\langle x, subj_2 \rangle \in \xi'_{\text{INHERIT}},\ \xi_2 = \langle obje_2, subj_2 \rangle \ \text{axiom 2}}{\langle obje_2, x \rangle}$$

$$\frac{\langle obje_2, x \rangle, \langle obje_2, y \rangle \in \left(\xi_{\text{CONTAIN}}\right)^k, \text{axiom 3}}{\langle y, x \rangle}$$

Thus, $\langle y, x \rangle$ was deduced.

Therefore, for x and y, conflict happens since $\langle x, y \rangle$ and $\langle y, x \rangle$ can both be deduced at the same time, so the sufficient condition is proved.

- the Proof of Necessary Condition

For any $\xi_1 = \langle subj_1, obje_1 \rangle$, $\xi_2 = \langle obje_2, subj_2 \rangle$, if there is conflict between ξ_1 and ξ_2.

Such there are no $x \in$ SUBJECT that:

$$\left(\langle x, subj_1 \rangle \in \left(\xi_{\text{INHERIT}} \right)^m \right) \wedge \left(\langle x, subj_2 \rangle \in \left(\xi_{\text{INHERIT}} \right)^n \right); m, n \geq 0; m, n \in N;$$

There are no interacting subjects between $subj_1$ and $subj_2$, so two rules are not conflicting.

Similarly, such there are no $y \in$ OBJECT that:

$$\left(\langle obje_1, y \rangle \in \left(\xi_{\text{CONTAIN}} \right)^j \right) \wedge \left(\langle obje_2, y \rangle \in \left(\xi_{\text{CONTAIN}} \right)^k \right); j, k \geq 0; j, k \in N;$$

Two rules are not conflicting.

Therefore, if two rules conflicted, the two conditions should be satisfied at the same time.

The necessary condition is proved.

Conflict-Related Rules.

Definition 8 conflict-related rules is rules that cause conflict in access control policy, written as $\Phi_{related}$. Conflict-related Rules is a set of rules:

$$\text{CONFLICTRULES} = \left\{ rule_1, rule_2, \ldots\ldots, rule_n \mid rule_i \in \text{POLICY}, 1 < i < n \right\} \tag{7}$$

The rules of conflict-related rules satisfy the two conditions as follows:

(1) There is conflict in CONFLICTRULES ;

(2)There would be no conflict, if one rule from CONFLICTRULES were erased.

From Theorem 1, we conclude that the rules cause conflict situation include three kinds of rules:

Definition 9. The two rules that have opposite actions, $\xi_1 = \langle subj_1, obje_1 \rangle$, $\xi_2 = \langle obje_2, subj_2 \rangle$, we denote as **rules have opposite actions, written as** $\Phi_{opposite}$.

Definition 10. The rules that can deduce subjects between rules having opposite actions ξ_1 and ξ_2 have inheritance relationship, we denote as **Subject overlap rules, written as** $\Phi_{subjects}$.

Definition 11. The rules that can deduce objects between rules having opposite actions ξ_1 and ξ_2 have contain relationship, we denote as **Object overlap rules, written as** $\Phi_{objects}$.

Theorem 2 $\Phi_{opposite}, \Phi_{subjects}, \Phi_{objects}$ **is a complete division of Conflict-related Rules (** $\Phi_{related}$ **).**

Proof:

$$\frac{r \in \Phi_{related},\quad definition8}{\Phi_{related} \Rightarrow conflict} \qquad \frac{\Phi_{related} \Rightarrow conflict,\quad theorem1}{\Phi_{related} = \{\xi_1, \xi_2\} \cup \Phi^1_{rules} \cup \Phi^2_{rules}}$$

$$which \quad \Phi^1_{rules} \Rightarrow condition1, \Phi^2_{rules} \Rightarrow condition2$$

$$\frac{\{\xi_1, \xi_2\}, definition9}{\{\xi_1, \xi_2\} = \Phi_{opposite}} \qquad \frac{\Phi^1_{rules}, definition10}{\Phi^1_{rules} = \Phi_{subjects}} \qquad \frac{\Phi^2_{rules}, definition11}{\Phi^2_{rules} = \Phi_{objects}}$$

$\Phi_{related} = \Phi_{opposite} \cup \Phi_{subjects} \cup \Phi_{objects}$ is proved.

4 A Conflict-Related Rules Conflict Detection Tool for AC Policy

This session implements a conflict-related rules conflict detection tool for AC policy, which could do detections of "conflict-related rules" when one access control rule is added by security administrator.

The tool is implemented basing on description logic.

Description logic is a kind of language represents that knowledge has grammar and semantics. Description logic is building on concept and relation (Relation, Role). Concept means the set of objects and relation means the binary relation between objects [2]. Description logic system consists of four basic parts: description logic language, description logic knowledge base, reasoning mechanism and query language supported by description logic system. Description logic language specifies the language of description logic. Description logic knowledge base is comprised of TBox (Terminological Box) and ABox (Assertion Box). TBox means terminology and terminology is the rules used for reasoning. ABox means assertion and assertion is the facts used for reasoning. Reasoning mechanism automatically does reasoning according to knowledge base. Query language supported by description logic system can query facts conforming to conditions.

Therefore, according to the grammar of TBox, axiom can be described as semantic model of access control policy and conditions for conflict-related rules are in session Ⅲ. The axiom will be used for reasoning and put into TBox. With the semantic model in TBox, the specified access control policy can be converted into instances in ABox and used as the facts of reasoning. Describing the "conditions for conflict-related rule" as axiom, by using the query language which is supported by the description logic system, "conflict-related rule" can be queried to complete the process of detection through reasoning.

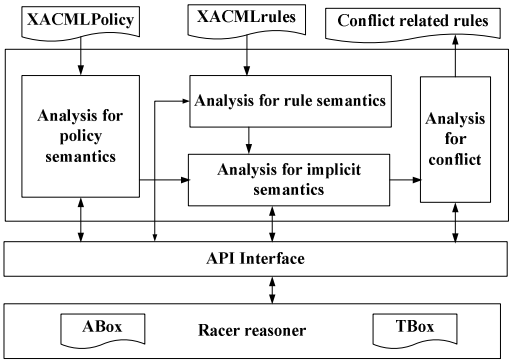

Fig. 1. The module structure of the conflict related rules detect tool for AC policy

The module structure of tool is presented as Fig.1.

The tool consists of four modules:

Analysis for policy semantics: read access control policy declaration described by XACML language, analyze its semantics, and put semantics in ABox through API interface.

Analysis for rule semantics: analyze rule semantics according to the input XACML rule, and then add the result in access control policy which is already put into ABox.

Analysis for implicit semantics: analyze implicit semantics in ABox according to the predefined SWRL rule, and then store it in the ABox.

Analysis for conflict: output conflict and conflict-related rules according to conflict reasoning rules which are described by predefined SWRL rule.

Implementation layer includes Racer reasoning machine and API interface. Racer reasoning machine is realized by adopting Racer 1.9.5 reasoning machine which includes ABox and TBox. TBox stores abstract model of access control rules and SWRL reasoning rules. ABox stores instances of access control rules. Through structures of nROL query language offered by JRacer, reports of "conflict-related rules" will query "conflict-related rules" that meet the conditions. The specific implementation of TBox construction, compiling of SWRL rule and "conflict-related rule" report are as follows.

The rules of reasoning the "conflict-related rules" described by SWRL (Semantic Web Rule Language) in the form of TBox axiom is as follows:

1) Subject(?sA)→has_Subject_Overlap(?sA, ?sA)
2) Resource(?rA)→has_ Resource_Overlap(?rA, ?rA)
3) Action(?aA)→has_ Action_Overlap(?aA, ?aA)
4) has_subSubject(?sA,?sB)∧has_subSubject(?sB,?sC)→ has_sub Subject (?sA, ?sC)
5) has_subSubject(?sA,?sB)→has_Subject_Overlap(?sA, ?sB)
6) has_Subject_Overlap(?sA,?sB)→has_Subject_Overlap(?sB, ?s A)
7) has_subResource(?rA, ?rB)∧has_subResource(?rB, ?rC) → has _subResource(?rA, ?rC)
8) has_subResource(?rA, ?rB)→ has_Subject_Overlap(?rA, ?rB)
9) has_Subject_Overlap(?rA,?rB)→has_Subject_Overlap(?rB, ?rA)
10) has_Subject(?pA, ?sA) ∧has_Subject(?pB, ?sB)∧ has_Subject _Overlap(?sA, ?sB) ∧ has_Resource(?pA, ?rA) ∧ has_Resource(? pB, ?rB)∧has_Resource_Overlap(?rA, ?rB)∧has_

Action(?p A, ?aA) ∧ has_Action(?pB, ?aB) ∧ has_Action_Overlap(?aA, ?aB) → has_Permissi on_Overlap (?pA, ?pB)

11) has_PermitA(?poA, ?pA)∧has_DenyB(?poB, ?pB) ∧ has_Per mission_Overlap(?pA, ?pB) →has_Policy_Conflict(?poA, ? poB)

12) has_PermitA(?poA, ?pA)∧has_DenyB(?poB, ?pB)∧ has_P ermission_Overlap(?pA, ?pB) →has_Policy_Conflict(?poB, ? poA)

13) has_PermitB(?poA, ?pA)∧has_DenyA(?poB, ?pB)∧has_Permission_Overlap(?pA, ?pB) → has_Policy_Conflict(?poA, ? poB)

14) has_PermitB(?poA,?pA) ∧ has_DenyA(?poB,?pB) ∧ has_Pe rmission_Overlap(?pA, ?pB) → has_Policy_Conflict(?poB, ?poA)

Where, 1)-6) describe conditions for overlap relationship, which includes subject overlap and object overlap. 1)-3) represent reflexivity of overlap relationship. 4) represents transitivity of overlap relationship. 6) represents symmetry of overlap relationship. 7)-9) represent overlap relationship between subjects, overlap relationship between objects, overlap relationship between actions respectively. 10) represents relations with overlapped subject, object and action. 11) -14) represent conflict.

The detection reports of "conflict-related rule" are input through queries on all instances satisfying "conflict-related rule condition". The queries are implemented by nRQL query language which is used by Racer.

5 Experiments

This section, we evaluate the correctness and effectiveness of our conflict-related rules detection tool. We use a policy of an information system to evaluate the correctness, and compare the response times of our tool and CPN based tool to evaluate the effectiveness.

The environment of experiments is: CPU: 2.93GHz，Memory: 4.00GB of RAM, operation system: Windows XP, reasoning engine: RacerPro 1.9.2 beta.

5.1 Correctness Analysis

We use the access control policy shown in introduction as input, which are written in XACML language and the output as Table 1 showed.

Using the tool to detect "conflict-related rules" in access control policy, the result is obtained after 1.4 second.

The output is represented as Table 1. As Table 1 shown, the "conflict-related rules" detected by the tool are divided into three types: rules of opposite actions, subject overlap rules, and object overlap rules.

Table 1. The result of the tool

Output type	output content
rule of opposite actions	"rule 1 and rule 3 have conflict"
subject overlap rule	"subject overlap rule is: rule 0"
object overlap rule	"object overlap rule is: rule 5"

The results of experiment show that: our conflict-related rules detection tool for AC policy is correctness: it detected all the conflict-related rules for access control policy, instead of detect the two rules with opposite actions, and this advantage makes our tool can help security administrator understand the information of the conflict situation more comprehensively.

5.2 Effectiveness Evaluation

Colored petri net (CPN) is an important method to represent and analyze the policy semantic [13]. In order to evaluate the efficiency of the proposed tool, we compare the response time of our tool and CPN based conflict detection tool. The test results represents as Fig.2.

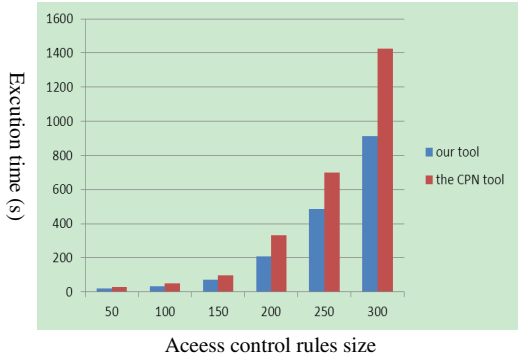

Fig. 2. Performance evaluation

The results of Fig.2 show that the response time of conflict-related rules detection tool is under 920s even the rule set size reaches 300, and the performance of our tool is obviously better than CPN based conflict detection tool's, because :

1) The CPN based method represent conflict-related rules and access control rules with place, transition, token and so on, which has many state results to long process time.

2) Our method represents conflict-related rules and access control rules with concepts and relations based on description logic, besides the tableau algorithm of the description logic has been optimized.

6 Conclusion

To detect the rules about conflict situation more comprehensively, this paper abstracted all the rules of the conflict situation as the concept of "conflict-related rules" and implemented a conflict detection tool. We analyzed the semantics of access control policy, and formally represented it with set theory; we defined the conflict-related rule for access control policy and deducted its extension. Based on the description logic, we realized the tool to detect conflict-related rules and we validated the correctness and

effectiveness of the tool in experiment with a policy of information system. The results of the correctness experiment showed the tool detected all the conflict-related rules for access control policy, which makes our tool can help security administrator understand the information of the conflict situation more comprehensively. And the results of the effectiveness experiment showed that our tool's response performance is better than VPN tools.

Our future work would pay attention to extend this tool to situations may have other types of conflict like SoD to detect the conflict-related rules.

Acknowledgment. We would like to express gratitude to associate Mr. Yang Bo, Chao Yuan, Junshun Hu, and Ms. Xue Qiu for many helpful discussions.

References

1. Sandhu, R., Ferraiolo, D.F., Kuhn, D.R.: The NIST Model for Role Based Access Control: Toward a Unified Standard. In: 5th ACM Workshop on Role Based Access Control, pp. 47–63. ACM Press (2000)
2. Doconta, M.C.: A guide to the future of xml, web services, and knowledge management. China Science and Technology Press, Beijing (2009)
3. Lupu, E.C., Sloman, M.: Conflicts in policy-based distributed systems management. IEEE Transactions on Software Engineering 25(6), 852–869 (1999)
4. Heilili, N., Chen, Y., et al.: An OWL-based approach for RBAC with negative authorization. Knowledge Science, Engineering and Management 4092, 164–175 (2006)
5. Lu, J., Li, R., Varadharajan, V., Lu, Z., Ma, X.: Secure Interoperation in Multi-domain Environments Employing UCON Policies. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 395–402. Springer, Heidelberg (2009)
6. Moon, C.-J., Paik, W., Kim, Y.-G., Kwon, J.-H.: The conflict detection between permission assignment constraints in role-based access control. In: Feng, D., Lin, D., Yung, M. (eds.) CISC 2005. LNCS, vol. 3822, pp. 265–278. Springer, Heidelberg (2005)
7. Shafiq, B., Joshi, J.B.D., Bertino, E., Ghafoor, A.: Secure interoperation in a multi-domain environment employing RBAC policies. IEEE Transactions on Knowledge and Data Engineering 17(11), 1557–1577 (2005)
8. Ni, Q.: Privacy-aware role-based access control. ACM Transactions on Information and System Security (TISSEC) 13(3), 1–31 (2010)
9. Huang, F., Huang, Z., Liu, L.: A DL-based method for access control policy conflict detecting. In: Internetware 2009, pp. 1–5. ACM, USA (2009)
10. Mohan, A., Blough, D.M.: Detection of Conflicts and Inconsistencies in Taxonomy-based Authorization Policies. In: 2011 IEEE International Conference on Bioinformatics and Biomedicine, GA, Atlanta, pp. 590–594.
11. Mansor, A.A., et al.: Policy-based approach to detect and resolve policy conflict for static and dynamic architecture. Journal of Theoretical and Applied Information Technology 37(2), 268–278 (2012)
12. Radi, A., et al.: On the three levels security policy comparison between SVM and decision trees. Journal of Theoretical and Applied Information Technology 35(1), 56–68 (2012)
13. Huang, H., Kirchner, H.: Formal specification and verification of modular security policy based on colored petri nets. IEEE Transactions on Dependable and Secure Computing 8(6), 852–865 (2011)