XACML Policy Inconsistency Analysis and Resolution

Teo Poh Kuang¹, Hamidah Ibrahim¹, Nur Izura Udzir¹ and Fatimah Sidi¹

¹Faculty of Computer Science and Information Technology, Universiti Putra Malaysia (UPM), 43400 UPM, Serdang, Malaysia

pohkuang1985@yahoo.com.my, {hamidah.ibrahim, izura, fatimah}@upm.edu.my

Keywords: XACML, Modality Inconsistency, Authorization Propagation, Inheritance Relationship, Inconsistency Resolution.

Abstract: Modality inconsistency is one of the security policy evaluation challenges, which arises because of the existence of both positive and negative authorizations for a given subject-object pair. An inconsistency analysis model is needed to discover inconsistency based on the inheritance relationship between concepts and resolved it by using predefined resolution rules. Previous studies handle modality inconsistency by providing the hierarchy of subjects and objects and simple conditions evaluation, like string equality matching. They do not identify modality inconsistency when a concept inherits conflicting decisions from its superclasses on the basis of the partially ordered structures obtained based on subject hierarchy, object hierarchy, and spatial hierarchy. An inconsistency analysis model is proposed in this paper to detect and resolve inconsistent policies during security policy evaluation. Our inconsistency analysis model analyzes all possible violations that might exist among security policies based on role hierarchy, object hierarchy, and spatial hierarchy. In addition, comparison with previous works shows that our inconsistency model is more effective in detecting inconsistency than the previous works.

1 INTRODUCTION

With the explosive growth of web technologies deployed on the Internet, collaborative applications such as distributed systems have been developed for data sharing and resource provisioning. Each organization designed security policies autonomously to serve their particular authority principle concern. Due to the dynamism and complexity of those collaborative applications, the number of policies can typically increase and suffer from inconsistencies which may affect the correctness of policy evaluation (Brodecki et al. 2012).

Authorization propagation policies explicitly specify what subjects can and cannot do. However, when a user sent a request to execute an action, if there is no explicit authorization specified for the user, there must be some way to propagate authorizations for the user (Jajodia et al. 2001). Modality inconsistency arises because of the existence of both positive and negative authorizations for a given subject-object pair in security policy evaluation. However, previous studies do not identify modality inconsistency when a concept inherits conflicting decisions from its superclasses on the basis of the partially ordered structures obtained based on subject hierarchy, object hierarchy, and spatial hierarchy. This may cause unauthorized disclosures of information or simply deny authorized access. Hence, an inconsistency analysis model is needed to discover modality inconsistency based on the inheritance relationship between concepts.

To tackle the limitations stated above, an inconsistency analysis model is proposed in this paper to detect and resolve inconsistent policies for security policy evaluation. All possible violations that might exist among security policies are identified based on subject, object, and spatial hierarchies. Our model is able to exploit the hierarchical structures in which policy elements (subject, object, and spatial) are organized. We implemented our algorithm in XACML since it is an Open Scripture Information Standard (OSIS) and the most widely used policy specification language for access control. In addition, comparison with previous works shows that our inconsistency model
is more effective in detecting inconsistency than the previous works.

The rest of the paper is organized as follows. The related policy inconsistency detection models are overviewed in Section 2. The motivation is investigated via a motivating scenario given in Section 3. Section 4 introduces the proposed inconsistency analysis model and algorithm. Comparison between the proposed model and prior models is conducted in Section 5. Section 6 reports the conclusion.

2 RELATED WORKS

With the increasing popularity of distributed systems and collaborative applications, there is a need to apply the inconsistency analysis in security policy evaluation. Traditional modality inconsistency is designed to detect two or more policies with modalities of opposite sign which refers to the same subject, action, and object (Lupu and Sloman, 1997). According to Borst and Akkermans (1997) the domain of elementary expressions that occurs in policies can be broadly classified into five categories: equality constraints, inequality constraints, real valued linear constraints, semantic constraints, and regular expression constraints. The domains of the attributes that appear in the above Boolean expressions belong to one of the following categories: string domain, real domain, integer domain, semantic domain, and tree domain. Hence, it is worth nothing that Lupu and Sloman (1997) evaluated the attributes of the policy that belong to a simple equality constraint.

Typically in a large distributed system, when a user sends a request to execute an action, if there is no explicit authorization specified for the user, there must be some way to propagate authorizations to the user (Jajodia et al., 2001). In other words, the authorization policies may be propagated according to the inheritance relationships between concepts which may cause inconsistencies. Mohan et al. (2011) presented strategies for modality conflict detection when access control policies are defined on taxonomies, which represent a hierarchical relationship (i.e., class-subclasses relationships) between object concepts only. While Brodecki et al. (2012), Damiani et al. (2006), and Jajodia et al. (2001) are able to analyise authorization propagation policy according to subsumption relationships between concepts on the basis of the partially ordered structures obtained based on subject hierarchy and object hierarchy. However, these works are limited to simple condition evaluation in which string equality matching is used.

Adi et al. 2009 argued that sometimes it is required to consider additional temporal as well as spatial constraints on the permission inheritance hierarchy in order to restrict policy permission. The senior role should be able to invoke the permissions of the junior role provided the senior role satisfies the spatio-temporal constraints of the inheritance hierarchy and also the spatio-temporal constraints needed to acquire the permissions of the junior role. In addition, complex condition elements such as semantic relationships between spatial elements, temporal elements or tree domain structures are necessary to take into account in the modality inconsistency analysis process.

In summary, none of the studies could provide an effective inconsistency analysis model which can derive an implicit authorization propagation policy based on subject hierarchy, object hierarchy, and spatial hierarchy. This will cause modality inconsistency could not be detected properly.

3 MOTIVATING SCENARIO

The following case study is used to present how policy inconsistency analysis worked through our proposed model. Figure 1 illustrated subject hierarchy (Hsub), object hierarchy (Hobj), and spatial hierarchy (Hspa) for university policies. Each of the access control policies specified on the superclasses is enforced for all of its subclasses. For instance, ExternalResource has two subclasses: OutSourceDeveloper and MasterStudent. If a Deny policy is defined on ExternalResource, thus all requests to the subclasses of ExternalResource should also be denied. We introduce the definition of policy and access request as follows:

Definition 1 An access control policy, $P$, is a tuple of the form:

$$P = (Decision, Subject, Resource, Action, Condition),$$

Definition 2 An access request, $Req$, is presented in the form:

$$Req = (Subject, Resource, Action, Condition).$$

$Subject$, $Resource$, and $Action$ are classical components of access control in the request and policy. However, $Condition$ is an optional parameter to further constrain the scope of the request or policy elements. The $Subject$ element identifies an individual user or a user role that can potentially
invoke an action in the system. The Resource element can be any object for subject to access (e.g. data or computer resources like Web-servers or database servers). The Action element represents any operation (e.g. delete or write a file) that can be applied to the resource in the policy. Finally, the Condition element is a Boolean expression that might involve environment context of evaluation. Typical environment elements that will be used in the condition are time (e.g. 2P.M. \( \leq \) time \( \leq \) 5P.M.) and spatial (e.g. location = Faculty Floor). Our work is able to cover the domain of elementary expressions which can be classified into four categories:

Category 1 One variable equality constraints, \( x = c \), where \( x \) is a variable and \( c \) is a constant.

Category 2 One variable inequality constraints, \( x > c \), where \( x \) is a variable, \( c \) is a constant, and \( > \in \{ <, \leq, \geq, >, \neq \} \).
Category 3 Semantic constraints, where \( f_{ab}(H_k, C, D) \), where \( H_k \) is a hierarchy, \( C \) and \( D \) are concepts belonging to \( H_k \).

Category 4 Compound Boolean expression constraints. This category combines the categories 1, 2, or 3 using the following logical operators: \&\& or \( \geq \) or \( \leq \).

The domains of the attributes that might appear in the above constraints belong to one of the following categories: string, real, integer, semantic or tree domain. A simple set of authorization based on Definition 1 is shown in Table 1, which contains six explicit policy authorizations. In Table 2, several requests based on the format defined in Definition 2 are presented. These requests will be used to clarify our model in detecting inconsistency.

4 THE PROPOSED INCONSISTENCY ANALYSIS MODEL

Figure 2 showed the overall general process flow of the proposed inconsistency analysis model that aims to detect all possible violations that might exist among the security policies. Each organization may have complex sets of security policies and such sets may consist of thousands of rules for security purposes. Thus, when a user sends a request to access the resources of an organization, the proposed inconsistency analysis model applies the authorization propagation policy derivation rule to retrieve the related explicit and implicit policies from the complex sets of thousands of rules. The authorization propagation policy derivation rule used in our model is as follow:

\[
\text{Subject}_{\text{req}} \leq \text{Subject}_{\text{pol}} \&\& \text{Object}_{\text{req}} \leq \text{Object}_{\text{pol}} \&\& \\
\text{Action}_{\text{req}} = \text{Action}_{\text{pol}} \&\& \text{Condition}_{\text{req}} \leq \text{Condition}_{\text{pol}}
\]

Figure 3 shows the inconsistency detection algorithm which helps to identify whether modality inconsistency exists among the authorization explicit and implicit policies. In our inconsistency detection algorithm, we compare the attribute of a request, \( req \), against the attribute of a policy, \( P \), to identify the inheritance relationships between them. At first, a user may send a request to access the resources of an organization. Subject hierarchy, object hierarchy and spatial hierarchy are applied as external resources with the purpose of deriving authorization propagation policy. Different paths in the subject, object and spatial hierarchies may propagate different authorizations to the user. The derivation rule stated that the explicit and implicit policies will be retrieved if the above derivation rule conditions are obeyed. We assumed that a concept is a specialization of another concept if it referred to a more specialized concept. \( \text{Sub}_{\text{req}} \leq \text{Sub}_{\text{pol}} \) means that the subject of the request, \( \text{Sub}_{\text{req}} \), is a specialization or equal concept to the subject of the policy, \( \text{Sub}_{\text{pol}} \).

For example, \( \text{ExternalResource} \) in \( P3 \) can be seen as a superclass of \( \text{MasterStudent} \) in \( Req3 \) based on the Figure 1(a). Hence the subject of \( Req3 \) is matched with the subject of \( P3 \), \( \text{Obj}_{\text{req}} \leq \text{Obj}_{\text{pol}} \) specifies that an object of the request, \( \text{Obj}_{\text{req}} \), is a specialization or equal concept to the object of the policy, \( \text{Obj}_{\text{pol}} \). For example, \( \text{TechnicalReport} \) in \( Req3 \) is a specialization of \( \text{Document} \) in \( P3 \) based on the Figure 1(b). In our proposed derivation rule, \( \text{Act}_{\text{req}} = \text{Act}_{\text{pol}} \) specifies that the action of the request, \( \text{Act}_{\text{req}} \), is an equal concept to the action of the policy, \( \text{Act}_{\text{pol}} \). The final condition, \( \text{Cond}_{\text{req}} \leq \text{Cond}_{\text{pol}} \) states that the condition of the request, \( \text{Cond}_{\text{req}} \), is a specialization or equal concept to the condition of the policy, \( \text{Cond}_{\text{pol}} \). For example, \( \text{Location: Faculty Floors} \) of \( Req3 \) is a subclass of \( \text{Location: Floor 5 of P1} \) based on the Figure 1(c).

The output of the inconsistency analysis process is a set of related policies and if it is identified that modality inconsistency occurs then further resolution rules are needed in order to resolve the modality inconsistency before a final authorization decision is returned. Four types of predefined resolution rules are applied in our work: "Permit-Overrides", "Deny-Overrides", "First-Applicable", and "Only-One-Applicable". We implemented our inconsistency model using Java 1.6.0_10 and evaluated the effectiveness of the inconsistency model on the sample XACML policies discussed in Damiani et al. (2006) which are designed for the university policies.

5 RESULT DISCUSSION

We compare the policies retrieved and the inconsistencies detected by our proposed model with the approaches proposed by Damiani et al. (2006), Adi et al. (2009) and Brodecki et al. (2012) based on the scenario discussed in Section 3. The results are summarized in Table 3.
A key observation of the results is that only our proposed model can detect modality inconsistency for most of the cases. Damiani et al. (2006) neglected the evaluation of all the requests since their method only focused on the subject and object authorization propagation. Although Adi et al. (2009) and Brodecki et al. (2012) improved the modality inconsistency algorithm by supporting condition attribute evaluation, but it is still limited to simple equality constraints. Thus, the requests which involved complex condition authorization propagation policy are not evaluated by both of the works.

The methods which do not consider complex condition constraint function will not retrieve the related explicit and implicit policies. This cause modality inconsistency could not be detected. The results are significant because the previous models are based on a simple inconsistency algorithm that relies on simple subject and object authorization propagation policy. The proposed model is able to detect the actual inconsistency based on the subject, object, and spatial hierarchies and produce accurate results than the previous models.
Table 3: Comparison Result between (a) Damiani et al. (2006), (b) Adi et al. (2009), (c) Brodecki et al. (2012), (d) Proposed Model.


<table>
<thead>
<tr>
<th>Request</th>
<th>Damiani et al., 2006</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit Rule</td>
</tr>
<tr>
<td>Req1</td>
<td>-</td>
</tr>
<tr>
<td>Req2</td>
<td>-</td>
</tr>
<tr>
<td>Req3</td>
<td>-</td>
</tr>
<tr>
<td>Req4</td>
<td>-</td>
</tr>
<tr>
<td>Req5</td>
<td>-</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Request</th>
<th>Adi et al., 2009</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit Rule</td>
</tr>
<tr>
<td>Req1</td>
<td>-</td>
</tr>
<tr>
<td>Req2</td>
<td>-</td>
</tr>
<tr>
<td>Req3</td>
<td>-</td>
</tr>
<tr>
<td>Req4</td>
<td>-</td>
</tr>
<tr>
<td>Req5</td>
<td>P6</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Request</th>
<th>Brodecki et al., 2012</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit Rule</td>
</tr>
<tr>
<td>Req1</td>
<td>-</td>
</tr>
<tr>
<td>Req2</td>
<td>-</td>
</tr>
<tr>
<td>Req3</td>
<td>-</td>
</tr>
<tr>
<td>Req4</td>
<td>-</td>
</tr>
<tr>
<td>Req5</td>
<td>P6</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Request</th>
<th>Our Proposed Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit Rule</td>
</tr>
<tr>
<td>Req1</td>
<td>-</td>
</tr>
<tr>
<td>Req2</td>
<td>-</td>
</tr>
<tr>
<td>Req3</td>
<td>-</td>
</tr>
<tr>
<td>Req4</td>
<td>-</td>
</tr>
<tr>
<td>Req5</td>
<td>P6</td>
</tr>
</tbody>
</table>

6 CONCLUSION

This paper addresses the significant need in detecting and resolving modality inconsistency for effective policy evaluation that could not or could partially enforce by previous methods. We have made two key contributions. First, we present details of the algorithm for modality inconsistency analysis in security policy evaluation. Second, we implemented our proposed algorithm in policy evaluation process and performed an extensive comparison with several existing models. Our experimental results show that our proposed algorithm achieved more effective results than the previous inconsistency detection models.

In this paper, several requests and simple authorization policies in XACML structure are used to motivate the needs to detect modality inconsistency in the process of policy evaluation. We believe our study is laying a foundation for effective policy evaluation in organization collaboration environments. The next stage of this work is to investigate the correctness and completeness of the proposed model by proving that it is able to identify all inconsistencies.

REFERENCES


