JACET

J. ADV COMP ENG TECHNOL, 4(4) Autumn 2018

An Effective Modality Conflict Model for Identifying Applicable Policies During Policy Evaluation

Teo Poh Kuang[1], Hamidah Ibrahim[2], Fatimah Sidi[3], Nur Izura Udzir[4]
1, 2, 3, 4 Computer Science Department, Universiti Putra Malaysia.
2(hamidah.ibrahim@upm.edu.my)

Received (2018-07-01) Accepted (2018-12-01)

Abstract: *Policy evaluation is a process to determine whether a request submitted by a user satisfies the access control policies defined by an organization. Modality conflict is one of the main issues in policy evaluation. Existing modality conflict detection approaches do not consider complex condition attributes such as spatial and temporal constraints. An effective authorization propagation rule is needed to detect the modality conflicts that occur among the applicable policies. This work proposes a modality conflict detection model to identify the applicable policies during policy evaluation, which supports an authorization propagation rule to investigate the class-subclass relationships of a subject, resource, action, and location of a request and a policy. The comparison with previous work is conducted, and findings show the solution which considers the condition attribute (i.e. spatial and temporal constraints) can affect the decision as to whether the applicable policies should be retrieved or not which further affect the accuracy of the modality conflict detection process. Whereas the applicable policies which are retrieved for a request can influence the detection of modality conflict among the applicable policies. In conclusion, our proposed solution is more effective in identifying the applicable policies and detecting modality conflict than the previous work .*

Keywords: *Access control policies, authorization propagation, effectiveness, modality conflict, policy evaluation, XACML.*

How to cite this article:
Teo Poh Kuang, Hamidah Ibrahim, Fatimah Sidi, Nur Izura Udzir. An Effective Modality Conflict Model for Identifying Applicable Policies During Policy Evaluation. J. ADV COMP ENG TECHNOL, 4(4) Autumn2018 : 255-266

I. INTRODUCTION

Policy evaluation is a process to determine whether a request satisfies the access control policies. A policy is said to be applicable to a request if the attribute values of the request are matched with the attribute values of the policy. With the increasing popularity of distributed systems and collaborative applications, there is a need to apply a conflict analysis method in policy evaluation. Modality conflict is an issue in policy evaluation which arises because of the existence of both positive and negative authorizations for a given subject-object[1] pair in policy evaluation.

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 for the user [12]. In other words, the authorization policies may be propagated according to the inheritance relationships between concepts which may cause inconsistencies modality conflict. Several works have been devoted to the topic of propagation of authorizations in

1 The terms object and resource are being used interchangeably in this paper.

distributed systems according to the inheritance relationships between concepts [1; 4; 6; 7; 12; 17; 24]. However, the concern of these works is only on the authorization propagation on the subject, resource, or action attributes, but not on the condition attributes. These works are limited to simple condition evaluation in which string equal function is used. In [1], the authors 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. In addition, complex condition elements such as semantic relationships between spatial or temporal elements are necessary to take into account in the modality conflict detection process.

This paper is an extension of our previous work in [14]. In [14] we proposed a heterogeneity policy evaluation engine called HXPEngine which aims at resolving syntactical and terminological conflicts between the attribute values of a request and a policy. While this paper focuses on the issues of modality conflicts. Hence, in this paper, we propose a modality conflict detection model based on eXtensible Access Control Markup Language (XACML) to identify the applicable policies, which relies on inheritance relationships between the attribute values of a request and a policy. The modality conflict detection model contains subject, resource, action, location hierarchies that supports a more adequate representation of their semantics. These hierarchies are formed based on the matching results collected from human experts. Each policy that is specified on a superclass is enforced for all of its subclasses. The authorization propagation rule can assist policy evaluation to investigate the class-subclass relationships between the attribute values of a request and a policy based on the hierarchical structures in which policy attributes (subject, resource, action, and condition) are organized, so that an authorization decision produces by the policy evaluation engine will not lead to unsafe authorization access. We mainly focus on a process before the actual policy evaluation is performed to assist the policy administrators during policy evaluation. Our solution attempts to filter out the irrelevant policies which help the policy administrators to resolve modality conflict among these potentially applicable policies. The modality conflict will be reported accordingly so that the policy administrators can resolve them according to their priority to better protect sensitive and private data.

Overall, the main contributions of this work are briefly described as follows:

1) A modality conflict detection model that aims to effectively identify the explicit and implicit policies during execution of a request is proposed.

2) The experimental results of the proposed solution are presented to prove its capability of retrieving the applicable policies and detecting modality conflict during policy evaluation.

The rest of this paper is organized as follows. Section 2 reviews the authorization propagation rules that are proposed by previous studies for detecting modality conflict. The limitations of each work are then identified. Section 3 presents in details the proposed modality conflict detection model and modality conflict detection algorithm to identify the applicable explicit and implicit policies during policy evaluation and if it is identified that modality conflict occurs then conflict resolution is needed in order to resolve the modality conflict before an authorization decision is returned. An illustrative example is presented based on the academy university domain in order to illustrate how modality conflict exists among access control policies when authorizations are being propagated. Section 4 presents the evaluation of the proposed solution by evaluating the performance of the modality conflict detection model and the results are compared to the previous work. The last section concludes this work.

II. RELATED WORKS

An access control policy determines the conditions to be fulfilled by a subject to gain access to a resource [22]. All the policies must be denied instance by instance if subjects or resources have no hierarchical structure, which will be burdensome in large systems. Hence in most realistic applications, subjects and resources are organized hierarchically. The authorizations can be derived according to the hierarchy-based derivation policies [8]. Thus, each node in the hierarchy will get additional positive or negative authorizations because of these hierarchy-based derivation policies. Consequently, the possibility to retrieve multiple applicable policies with different effects to an authorization decision increased which may lead to modality conflict. Therefore, authorization propagation is a convenient and easy way to specify implicit policies, but it can result in unforeseen conflicts [13]. Hence, it is necessary to detect and resolve the modality conflict when both a denial and a permission are specified among explicit and implicit policies.

Past works [3; 9; 10; 16; 19; 20; 21] supported traditional modality conflict detection which has no hierarchical structure for organizing subject, action, resource, and condition. Therefore, none of these works provide an effective conflict detection method. This caused modality conflict could not be detected properly.

Several studies focused on the design, implementation and evaluation of a mechanism that can be used by policy administrators to proactively detect conflict policies among a set of policies in a policy database [1; 6; 11; 13; 18; 23; 24; 25; 26; 27]. Nevertheless, these works mainly focused on the modality conflict detection and resolution among the attribute values of policies once a new party joined the collaboration. The conflict analysis is generally much slower during policy design time especially for organization with policies of larger sizes [17].

A number of works have been devoted to the topic of propagation of authorizations based on the inheritance relationships between concepts [1; 4; 6; 7; 12; 17; 24].

The authors in [4] assume that the access permissions given to a role subsumed the access permissions given to all roles with a lower position in the hierarchy of an organization. The same concept is applied to object hierarchy. Based on the authorization propagation concept, the implicit permissions can be derived by inheriting permissions that are propagated to the requested node from the parent nodes. Thus, authorizations of opposite sign on the parent nodes may be propagated to the requested node which may cause inconsistencies.

The authors in [12] present a unified framework which allows the specification of both positive and negative authorizations and incorporated notions of authorization derivation, conflict resolution, and authorization decision strategies by exploiting the hierarchical structures of attributes (roles, user group, and resources). The authorizations of a node are propagated to all its descendants in the hierarchy.

The authors in [7] exemplify modality conflict arising from "part-of" relations. The modality conflict identified by this work can be resolved by considering the hierarchy concept used for propagating authorizations. The specificity principle being applied is based on the notion of domain nesting principle whenever this relationship exists. The authors in [17] applied descending propagation by evaluating the parent requested resource node and if the response is "Deny" for any of the parent nodes, the authorization decision is returned as "Deny". This work applied descending propagation by evaluating the child nodes which have authorization decision different from the requested resource node. The authors in [6] proposed an algorithm to discover modality conflict among two policies with opposite authorization decisions when descending propagation is applied with the knowledge of the hierarchy of subjects and resources. In [24], the authors proposed a novel method for detecting modality conflict among the access control policies when the concepts of role hierarchy and permission inheritance are introduced in the access control model.

The concern of the above works is only on the authorization propagation on the subject, resource, and action attributes, but not on the condition attributes and thus affects the result of authorization decision. These works are limited to simple condition evaluation in which string equal function is used. This caused modality conflict could not be detected properly.

The authors in [1] 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. Moreover, the collaborative nature of a distributed environment requires the specification of condition such as contextual constraints in the access control policy to ensure adequate protection of services and resources [2]. As context information get involved, the authorization propagation no longer depends on subject, resource, and action inheritance relationships, it also depends on the context information. Thus, spatial constraints, i.e. the requestor's location information and temporal constraints, i.e. the requestor's access time have to take into consideration in an access control policy when determining an access to services and resources as well as in the modality conflict detection process.

III. THE MODALITY CONFLICT DETECTION PROCESS

In distributed computing systems, the policies can grow very large and suffer from conflict which heavily influences the correctness and completeness in retrieving the applicable policies for a particular request. Thus, detecting and resolving the modality conflict is a non-trivial task because it involves identification of applicable policies and detecting modality conflict among them. In this section, we present in details our proposed modality conflict detection model for identifying the applicable policies and detecting modality conflict among the applicable policies.

The definition of modality conflict is as follows:

Definition 1 Modality conflict is inconsistency in the restriction policy specification, which arises when two policies Pol1 and Pol2 with modalities of opposite effects for the same subject, resource, action, and condition, i.e.:

$$Pol1 \equiv (Permit, Subject_{pol}, Resource_{pol}, Action_{pol}, Condition_{pol})$$

$$Pol2 \equiv (Deny, Subject_{pol}, Resource_{pol}, Action_{pol}, Condition_{pol})$$

During modality conflict detection process, it is possible that more than one applicable policy are retrieved with modalities of opposite effects. The modality conflict could arise from semantic relationships between concepts that cannot be detected simply by looking at the terminology structure of the terms. According to work in [5], when an access right for a subject on a resource is explicitly specified this is referred to as explicit authorization. While an access right for a subject on a resource can be implicitly derived from other explicit authorizations, referred to as implicit authorization. Authorizations are propagated along partially ordered structures obtained by classifying subject, resource, action, and condition attributes. All the authorizations of a node are propagated to its child nodes. If this is not done, then a user may be permitted to access a resource if he selects one access hierarchy path, while he may be denied access through another path [17]. For instance, if a "Permit" policy is defined on Student, thus all the requests to the subclasses of Student such as Undergraduate Student should also be permitted. However, if the subclasses of Student have opposite mode different from Student, the subclasses will inherit the conflicting mode from Student. Thus, we proposed a modality conflict detection model in order to identify all possible explicit and implicit applicable policies for a particular request before the modality conflict detection process is embarked. Fig. 1 shows the overall general process flow of the proposed modality conflict detection model.

In a large authorization system, there may be multiple authorities who specify the sets of policies and such sets may consist of thousands of rules for security purposes. When a user sends a request to access the resources of an organization, the authorization module will determine which policy is applicable to the particular request. There can be multiple policy sets and multiple policies in each set applicable to a single request. Even within each policy there can be multiple rules which are applicable to the request. These applicable policies can have a different or even conflicting authorization decision for the request.

Our modality conflict detection model contains subject, resource, action, and location which are organized into hierarchies, supporting a more adequate representation of their semantics. From the authorizations that are explicitly specified, implicit authorizations are automatically derived by the model.

Authorizations are automatically propagated along subject, resource, action, and location hierarchies and the authorization flows are always from the parent towards its child nodes. That is, a policy defined on a node should be enforced for all its children as well. In other words, each node either has its own policies or inherits them from its parent. Hence, implicit policies can be derived based on these hierarchies. However, policies for a concept can differ from its children and they can be in a conflict authorization decision. Our work provides an authorization propagation rule in order to investigate the class-subclass relationships of a subject, resource, action and location of a request and a policy before the applicable policy is identified. The proposed authorization propagation rule used in our model is as follow:

$$Subject_{req} \leq Subject_{pol} \;\&\&\; Resource_{req} \leq Resource_{pol} \;\&\&\; Action_{req} \geq Action_{pol} \;\&\&\; Location_{req} \leq Location_{pol} \quad (1)$$

where $Subject_{req}$ ($Subject_{pol}$) is the subject of the request (policy, respectively),

$Resource_{req}$ ($Resource_{pol}$) is the resource of the request (policy, respectively),

$Action_{req}$ ($Action_{pol}$) is the action of the request (policy, respectively),

$Location_{req}$ ($Location_{pol}$) is the location of the request (policy, respectively).

The concept is classified as a partial ordered

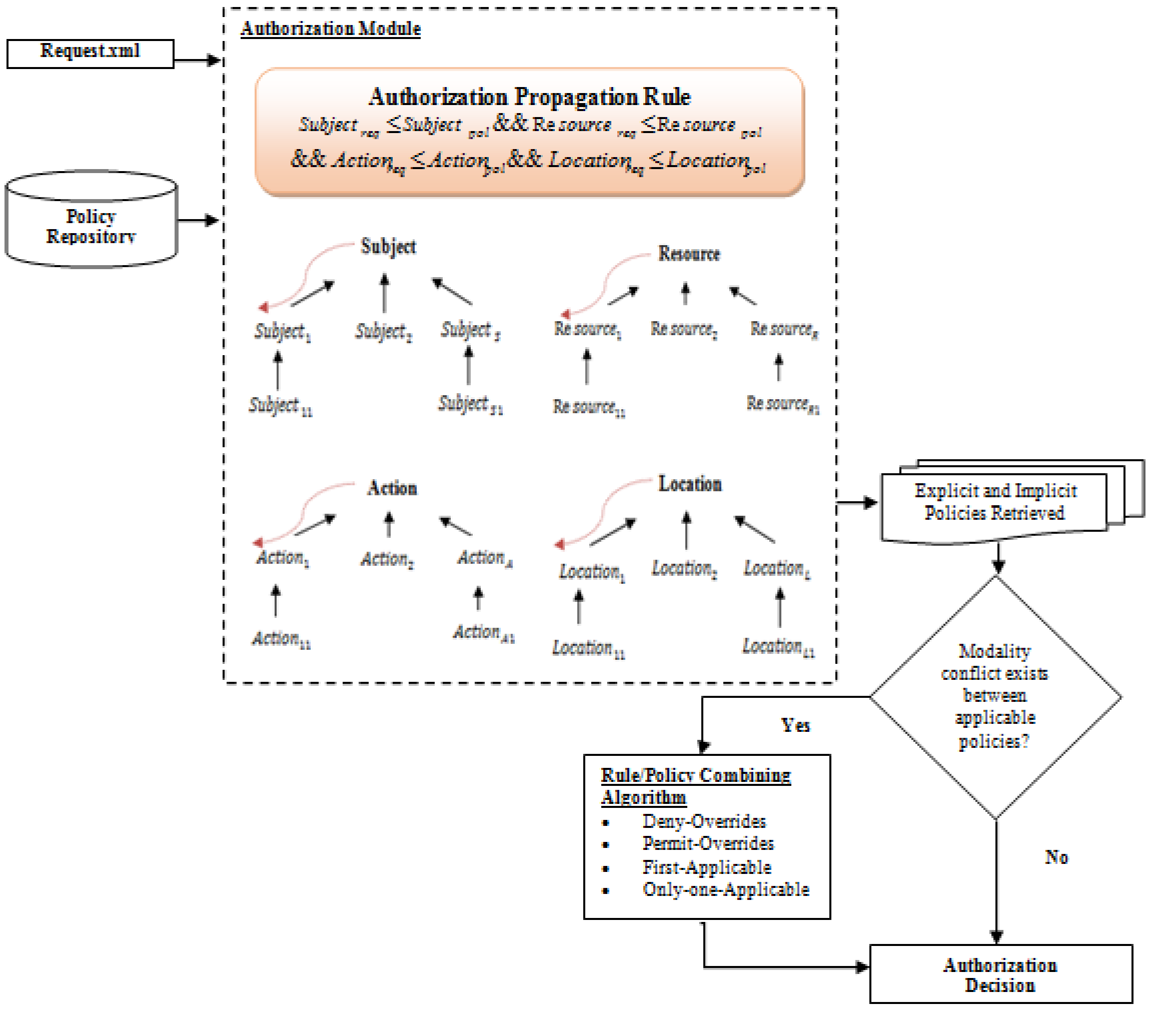
Request.xml
Policy Repository
Authorization Module
Authorization Propagation Rule
$Subject_{req} \leq Subject_{pol}$ && $Resource_{req} \leq Resource_{pol}$ && $Action_{req} \leq Action_{pol}$ && $Location_{req} \leq Location_{pol}$
Subject
$Subject_1$ $Subject_2$ $Subject_S$
$Subject_{11}$ $Subject_{S1}$
Resource
$Resource_1$ $Resource_2$ $Resource_R$
$Resource_{11}$ $Resource_{R1}$
Action
$Action_1$ $Action_2$ $Action_A$
$Action_{11}$ $Action_{A1}$
Location
$Location_1$ $Location_2$ $Location_L$
$Location_{11}$ $Location_{L1}$
Explicit and Implicit Policies Retrieved
Modality conflict exists between applicable policies?
Yes
No
Rule/Policy Combining Algorithm
Deny-Overrides
Permit-Overrides
First-Applicable
Only-one-Applicable
Authorization Decision

Fig. 1. The modality conflict detection model.

structure where an attribute value of a request, av_{req} is a specialization of an attribute value of a policy, av_{pol} if and only if $av_{req} \leq av_{pol}$, where ≤ represents the subsumption operator. This structure can ground the permission inheritance of the authorization propagation, i.e. rights assigned to concepts can be inherited by subsumed concepts [7]. Fig. 2 presents the modality conflict detection algorithm.

The attribute value of a request, *Req*, is compared to the attribute value of a policy, *P*, to identify the inheritance relationships between them. Based on the proposed authorization propagation rule, the explicit and implicit policies will be retrieved if the above propagation rule conditions are obeyed. *N-gram* and WordNet as an external thesaurus are applied in this work in matching the attribute values of a request and a policy [14]. *N-gram* is utilized to resolve the syntactic variations while WordNet is utilized to resolve the terminological variations. Our work assumes that semantic relationships (i.e. class-subclass) exist among these concepts. The underlying idea is that the parent-child relationship implies that one rule could be a restriction of the other and this would be more helpful than the sibling relationship [15]. The modality conflict detection algorithm will further check whether modality conflict exists among the applicable policies and if it occurs then conflict resolution is needed to resolve the conflict before an authorization decision is returned. XACML defines four types of predefined combining algorithm to automatically resolve modality conflict namely: "Permit-Overrides", "Deny-Overrides", "First-Applicable", and "Only-One-Applicable".

1. Illustrative Example

Fig. 3 depicts an example of a subject hierarchy (H_{sub}), resource hierarchy (H_{res}), action hierarchy (H_{act}), and location hierarchy (H_{loc}) for the university policies. These hierarchies are formed based on the results collected from human experts. Each policy that is specified on a superclass is enforced for all of its subclasses. For instance, the superclass *Student* has two subclasses, namely: *Graduate Student* and *Undergrad*. If a "Permit" policy is defined on *Student*, thus all requests to the subclasses of *Student* should also be permitted.

The policies and requests based on the university domain are used in this illustrative example. Table I presents the university policies used in the illustrative example while Table II presents the requests for the illustrative example.

Consider the subject, resource, action, and location hierarchies in Fig. 3 that are applied in deriving the implicit policies from the explicit policies for each of the following request:

i. For *Req1*, *pol2*$_{implicit}$ the implicit policies, and *AssociateProf*, *Grades*, *Assign*, and *GraduateSchool* in *Req5* are matched to *AssociateProfessor*, *Grades*, *Assign*, and *School* in *Pol5*, respectively. *Pol3*$_{implicit}$ as presented in Table III, are derived from the explicit policies, *Pol2* and *Pol3*, respectively. *Undergraduate Student* in *Req1* is matched to *Undergrad* based on the *N-gram* similarity measure and WordNet and *Undergrad* is a child node of *Student* in the subject hierarchy as shown in Fig. 3(a). Hence, *Undergraduate Student* in *Req1* is a subclass of *Student* in *Pol2*. *Teaching Course* in *Req1* is a subclass of *Course* in *Pol2* based on Fig. 3(b). *View* in *Req1* is matched to *View* in *Pol2*. *University Department* in *Req1* is a subclass of *Department* in *Pol2* since *University Department* is a child node of *Department* based on Fig. 3(d). Using similar matching process as explained above, *Pol3*$_{implicit}$ is derived from *Pol3*.

ii. For *Req2*, the implicit policy, *Pol1*$_{implicit}$ as presented in Table IV is derived from the explicit policy, *Pol1*. Based on Fig. 3, *N-gram* similarity measure and WordNet, *ResearchAssistant*, *ExternalGrades*, *Assign*, and *Institute* in *Req2* are matched to *RA*, *ExternalGrades*, *Assign*, and *Association* in *Pol1*, respectively.

iii. For *Req3*, the implicit policies, *Pol4*$_{implicit}$ and *pol5*$_{implicit}$ are derived from the explicit policies, *Pol4* and *Pol5*, respectively. Table V presents the implicit policies which are derived from *Pol4* and *Pol5* based on *Req3*. Based on Fig. 3, *N-gram* similarity measure and WordNet, *AssociateProfessor*, *InternalGrades*, *Assign*, and *GraduateSchool* in *Req3* are matched to *AssociateProfessor*, *Grades*, *Assign*, and *GraduateSchool* in *Pol4*, respectively. While *AssociateProfessor*, *InternalGrades*, *Assign*, and

```
Input:
Req: A request from a user in $Org_A$
$P_i$: A set of policies, { $P_1$, $P_2$, …, $P_m$ } in $Org_B$
Hierarchy: Subject hierarchy, $H_{sub}$, resource hierarchy, $H_{res}$, action hierarchy, $H_{act}$, location hierarchy, $H_{loc}$
Output:
Conflict/No Conflict

For each attribute value of Req ( $Subject_{req}$ , $Resource_{req}$ , $Action_{req}$ , $Location_{req}$ ) and $P_i$ ( $Subject_{pol}$ , $Resource_{pol}$ , $Action_{pol}$ , $Location_{pol}$ )
  Apply N-gram similarity measure and WordNet
  If ( $Subject_{req} \le Subject_{pol}$ ) && ( $Resource_{req} \le Resource_{pol}$ ) && ( $Action_{req} \le Action_{pol}$ ) && ( $Location_{req} \le Location_{pol}$ )
      Then retrieved $P_i$ and store it into an array, $array_{app\_pol}$
  Else check next policy
End For

For each applicable policy of $array_{app\_pol}$ , compare the effect of the applicable policy and check whether modality conflict exists
  If $Effect_{P_i} \ne Effect_{P_j}$ in policy set level
      Return Conflict
      Resolve modality conflict based on the policy combining algorithm as stated in the policy
  ElseIf $Effect_{P_i} \ne Effect_{P_j}$ in policy level
      Return Conflict
      Resolve modality conflict based on the rule combining algorithm as stated in the rule
  Else Return No Conflict
End For
```

Fig. 2. The modality conflict detection algorithm.

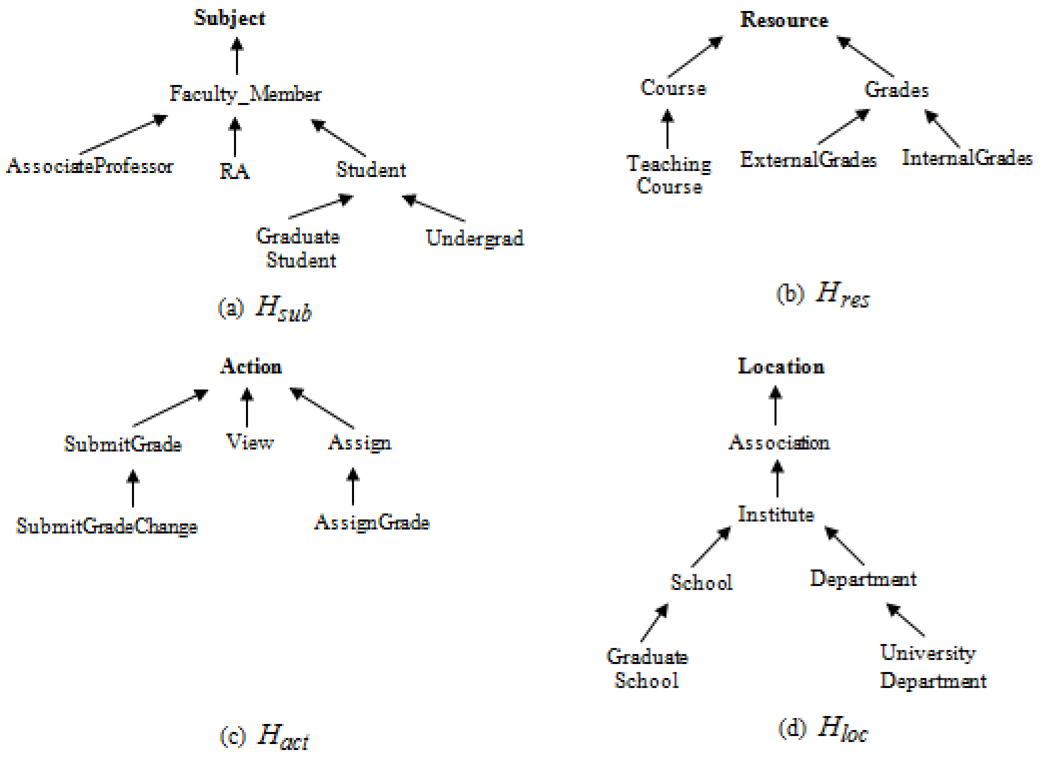

Fig. 3. (a) Subject Hierarchy for University Policy, , (b) Resource Hierarchy for University Policy, , (c) Action Hierarchy for University Policy, , and (d) Location Hierarchy for University Policy, .

GraduateSchool in *Req3* are matched to *Faculty_Member*, *Grades*, *Assign*, and *School* in *Pol5*, respectively. *Faculty_Member* is a superclass of *AssociateProfessor* and *School* in *Req4* is a superclass of *GraduateSchool* in *Pol4* which violate the concept of authorization propagation based on the inheritance relationships. Thus, no implicit policy is derived from *Pol4* based on *Req4*. As for *Pol5*, each attribute value of *Req4* is an exact match to its equivalent attribute value in *Pol5*. In this case, no implicit policy is derived from *Pol5* based on *Req4*.

v. For *Req5*, the implicit policies, $Pol4_{implicit}$ and $Pol5_{implicit}$ as presented in Table VI, are derived from the explicit policies, *Pol4* and *Pol5*, respectively. Based on Fig. 3 and *N-gram* similarity measure, *AssociateProf*, *Grades*, *Assign*, and *GraduateSchool* in *Req5* are matched to *AssociateProfessor*, *Grades*, *Assign*, and *GraduateSchool* in *Pol4*, respectively. While *AssociateProf*, *Grades*, *Assign*, and *GraduateSchool* in *Req5* are matched to *AssociateProfessor*, *Grades*, *Assign*, and *School* in *Pol5*, respectively.

iv. For *Req6*, the explicit policy, *Pol4* is retrieved and the implicit policy is derived from *Pol5*. *AssociateProfessor* in *Pol4* cannot be propagated to *Faculty_Member* in *Req6* since *Faculty_Member* is a superclass of *AssociateProfessor* based on Fig. 3(a) and *School* in *Req6* is a superclass of *GraduateSchool* in *Pol4* based on Fig. 3(d) which violate the concept of authorization propagation based on the inheritance relationships. Thus, no implicit policy is derived from *Pol4* based on *Req6*.

Table VII presents the implicit policy, $Pol5_{imlpicit}$ which is derived from *Pol5* based on *Req6*. Based on Fig. 3 and *N-gram*, *Faculty_Member*, *Grades*, *AssignGrade*, and *School* in *Req6* are matched to *Faculty_Member*, *Grades*, *Assign*, and *School* in *Pol5*, respectively.

v. For *Req4*, the proposed solution retrieved the explicit policies, *Pol4* and *Pol5*. *AssociateProfessor* in *Pol4* cannot be propagated to *Faculty_Member* in *Req4* since *Faculty_Member* is a superclass of *AssociateProfessor* and *School* in *Req4* is a superclass of *GraduateSchool* in *Pol4* which violate the concept of authorization propagation based on the inheritance relationships. Thus, no implicit policy is derived from *Pol4* based on *Req4*.
As for *Pol5*, each attribute value of *Req4* is an exact match to its equivalent attribute value in *Pol5*. In this case, no implicit policy is derived from *Pol5* based on *Req4*.

vi. For *Req5*, the implicit policies, $Pol4_{implicit}$ and $Pol5_{implicit}$ as presented in Table VI, are derived from the explicit policies, *Pol4* and *Pol5*, respectively. Based on Fig. 3 and *N-gram* similarity measure, *AssociateProf*, *Grades*, *Assign*, and *GraduateSchool* in *Req5* are matched to *AssociateProfessor*, *Grades*, *Assign*, and *GraduateSchool* in *Pol4*, respectively. While *AssociateProf*, *Grades*, *Assign*, and *GraduateSchool* in *Req5* are matched to *AssociateProfessor*, *Grades*, *Assign*, and *School* in *Pol5*, respectively.

vi. For *Req6*, the explicit policy, *Pol4* is retrieved and the implicit policy is derived from *Pol5*. *AssociateProfessor* in *Pol4* cannot be propagated to *Faculty_Member* in *Req6* since *Faculty_Member* is a superclass of *AssociateProfessor* based on Fig. 3(a) and *School* in *Req6* is a superclass of *GraduateSchool* in *Pol4* based on Fig. 3(d) which violate the concept of authorization propagation based on the inheritance relationships. Thus, no implicit policy is derived from *Pol4* based on *Req6*.

Table VII presents the implicit policy, which is derived from Pol5 based on Req6. Based on Fig. 3 and N-gram, Faculty_Member, Grades, AssignGrade, and School in Req6 are matched to Faculty_Member, Grades, Assign, and School in Pol5, respectively.

IV. RESULTS AND DISCUSSION

We compared the applicable policies identified and the modality conflict detected among the applicable policies by our proposed solution to those obtained by Sun's XACML implementation [21] and the human experts for each request.

We choose Sun's XACML implementation in our comparison for two reasons. First, it is the first and the most widely deployed XACML

TABLE I
THE UNIVERSITY POLICIES FOR THE ILLUSTRATIVE EXAMPLE

Policy Combining Algorithm	Rule Combining Algorithm	Policy No.	Effect	Subject	Resource	Action	Condition
Permit-Overrides	Permit-Overrides	*Pol*1	Permit	RA	ExternalGrades	Assign ∨ View	(Location = Association) ∧ (Time ≥ 12P.M. ∧ Time ≤ 2P.M.) ∧ (Email = upm.edu.my)
Permit-Overrides	Permit-Overrides	*Pol*2	Deny	Student	Course	Assign ∨ View	(Location = Department) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.) ∧ (Email = upm.edu.my)
		*Pol*3	Permit	Undergrad	Course	View	(Location = Department) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.) ∧ (Email = upm.edu.my)
Deny-Overrides	Permit-Overrides	*Pol*4	Permit	AssociateProfessor	Grades	Assign ∨ View ∨ SubmitGrade ∨ SubmitGradeChange	(Location = GraduateSchool) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)
	Permit-Overrides	*Pol*5	Deny	Faculty_Member	Grades	Assign ∨ View	(Location = School) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)

TABLE II
THE REQUESTS FOR THE ILLUSTRATIVE EXAMPLE

Request No.	Subject	Resource	Action	Condition
*Req*1	Undergraduate Student	Teaching Course	View	(Location = University Department) ∧ (Time = 12.30P.M.) ∧ (Email = gs23442@upm.edu.my)
*Req*2	ResearchAssistant	ExternalGrades	Assign	(Location = Institute) ∧ (Time = 1.30P.M.) ∧ (Email = gs23442@upm.edu.my)
*Req*3	AssociateProfessor	InternalGrades	Assign	(Location = GraduateSchool) ∧ (Time = 12.30P.M.) ∧ (Email = gs23442@upm.edu.my)
*Req*4	Faculty_Member	Grades	View	(Location = School) ∧ (Time = 12.30P.M.)
*Req*5	AssociateProf	Grades	Assign	(Location = GraduateSchool) ∧ (Time = 12.30P.M.)
*Req*6	Faculty_Member	Grades	AssignGrade	(Location = School) ∧ (Time = 12.30P.M.)

TABLE III
THE IMPLICIT POLICIES DERIVED FROM *POL2* AND *POL3* BASED ON THE *REQ1*

Policy No.	Effect	Subject	Resource	Action	Condition
$Pol2_{implicit}$	Deny	Undergraduate Student	Teaching Course	View	(Location = University Department) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.) ∧ (Email = upm.edu.my)
$Pol3_{implicit}$	Permit	Undergraduate Student	Teaching Course	View	(Location = University Department) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.) ∧ (Email = upm.edu.my)

TABLE IV
THE IMPLICIT POLICY DERIVED FROM *POL1* BASED ON THE *REQ2*

Policy No.	Effect	Subject	Resource	Action	Condition
$Pol1_{implicit}$	Permit	ResearchAssistant	ExternalGrades	Assign	(Location = Institute) ∧ (Time ≥ 12P.M. ∧ Time ≤ 2P.M.) ∧ (Email = upm.edu.my)

TABLE V
THE IMPLICIT POLICIES DERIVED FROM *POL4* AND *POL5* BASED ON THE *REQ3*

Policy No.	Effect	Subject	Resource	Action	Condition
$Pol4_{implicit}$	Permit	AssociateProfessor	InternalGrades	Assign	(Location = GraduateSchool) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)
$Pol5_{implicit}$	Deny	AssociateProfessor	InternalGrades	Assign	(Location = GraduateSchool) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)

TABLE VI
THE IMPLICIT POLICIES DERIVED FROM *POL4* AND *POL5* BASED ON THE *REQ5*

Policy No.	Effect	Subject	Resource	Action	Condition
$Pol4_{implicit}$	Permit	AssociateProf	Grades	Assign	(Location =GraduateSchool) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)
$Pol5_{implicit}$	Deny	AssociateProf	Grades	Assign	(Location = GraduateSchool) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)

TABLE VII
THE EXPLICIT POLICY AND IMPLICIT POLICY DERIVED FROM *POL5* BASED ON THE *REQ6*

Policy No.	Effect	Subject	Resource	Action	Condition
$Pol5_{implicit}$	Deny	Faculty_Member	Grades	AssignGrade	(Location = School) ∧ (Time ≥ 12P.M. ∧ Time ≤ 1P.M.)

evaluation engine and has become the industrial practice [16]. Second, the previous works [3; 16;19] selected Sun's XACML implementation for their results comparison since Sun's XACML implementation is an open source. These works focused on the efficiency of their engine by reducing the processing time while the results obtained are the same as compared to Proctor [21]. While our work focuses on the accuracy of identifying the applicable policies and detecting modality conflict among the applicable policies.

To provide a ground for evaluating the quality of the matching results, the results produced by our proposed solutions are compared to the human experts' results. The task was first conducted manually by three professional human experts who are either familiar with database management or English linguistics. Table VIII presents the explicit and implicit applicable policies retrieved for each of the request and the modality conflict detected at different policy level.

The proposed solution is able to retrieve both explicit and implicit applicable policies which are the same as the applicable policies retrieved by the human experts for all the requests in this illustrative example. *Req1*, *Req3* and *Req5* are the requests in which modality conflict is detected by the human experts as well as our proposed solution.

Consider the subject, resource, action, and location hierarchies in Fig. 3 that are applied in identifying the applicable policies which are shown in Table VIII for each of the following request:

i. For *Req1*, the proposed solution applied the rule combining algorithm, "Permit-Overrides" to resolve the modality conflict, in which "Permit" is returned as the authorization decision. For *Req3* and *Req5*, the proposed solution chooses the policy combining algorithm, "Deny-Overrides" to resolve the modality conflict at the policy set level. Thus, "Deny" is returned as the authorization decision. The modality conflict is not detected by Sun's XACML implementation for *Req1*, *Req3*, and *Req5*, thus, "N/A" is returned as the authorization decision.

ii. For *Req2* and *Req6*, there is no modality conflict detected by the proposed solution. Thus, the effect of $Pol1_{implicit}$, "Permit" is returned as the authorization decision for *Req2* while the effect of $Pol5_{implicit}$, "Deny" is returned as the authorization decision for *Req6*. "N/A" is returned as the authorization decision by Sun's XACML implementation since there is no applicable policy retrieved for *Req2* and *Req6*.

iii. For *Req4*, both the Sun's XACML implementation and the proposed solution retrieved the explicit policy, *Pol5* and the effect of *Pol5*, "Deny" is returned as the authorization decision.

Overall, the proposed solution successfully produced accurate results compared to the Sun's XACML implementation since the Sun's XACML implementation does not investigate the authorization propagation along the subject, resource, action and location hierarchies. Therefore, we can conclude that the proposed solution is better compared to the Sun's XACML implementation.

V. CONCLUSION

Policy evaluation has received considerable attention to accommodate the security requirements covering large, open, distributed computing environments. This research addresses the significant need in identifying the applicable policies and detecting modality conflict for XACML policy evaluation.

Our modality conflict model supports the authorization propagation rule which explores inheritance relationships of a subject, resource, action, and condition which enables the applicable policies to be retrieved for a given request. We present the algorithm for identifying applicable policies and detecting modality conflict based on the proposed authorization propagation rule.

Several requests and policies for a university in XACML structure are used to motivate the needs to identify the applicable policies and detect modality conflict in the process of policy evaluation. The analysis results show that our proposed solution achieved more effective results in retrieving the applicable policies and in detecting the modality conflict as compared to the previous work. This indicates that our proposed solution is better than the Sun's XACML implementation in policy evaluation. The next stage of this work is to investigate the effectiveness of the proposed model on real XACML policies designed for the university, conference management, and health care domain with respect to identify the applicable policies and detect modality conflict. This will be performed by analyzing decisions from human experts (professional groups) whom are familiar with policies.

REFERENCES

1. Almutairi, A., Sarfraz, M., Basalamah, S., Aref, W., and Ghafoor, A. 2012. A Distributed Access Control Architecture for Cloud Computing. IEEE Software, 29(2), pp. 36-44.

2. Ammar, N., Malik, Z., Bertino, E., & Rezgui, A. 2015. XACML Policy Evaluation with Dynamic Context Handling. IEEE Transactions on Knowledge and Data Engineering, 27(9), pp. 2575-2588.

3. Bertino, E., Ghinita, G., and Kamra, A. 2011. Access Control for Databases: Concepts and Systems. Foundations and Trends in Databases, 3(1-2), pp. 1-148.

4. Brodecki, B., Szychowiak, M., and Sasak, P. 2012. Security Policy Conflicts in Service Oriented Systems. New Generation Computing, 30(2-3), pp. 215-240.

5. di Vimercati, S. D. C., Foresti, S., Jajodia, S., and Samarati, P. 2007. Access Control Policies and Languages in Open Environments. Secure Data Management in Decentralized Systems, pp. 21-58.

6. Hu, H., Ahn, G., and Kulkarni, K. 2013. Discovery and Resolution of Anomalies in Web Access Control Policies. IEEE Transactions on Dependable and Secure Computing, 10(6), pp. 341-354.

7. Jajodia, S., Samarati, P., Sapino, M. L., and Subrahmanian, V. 2001. Flexible Support for Multiple Access Control Policies. ACM Transactions on Database Systems (TODS), 26(2), pp. 214-260.

8. Lin, D., Rao, P., Ferrini, R., Bertino, E., and Lobo, J. 2013. A Similarity Measure for Comparing XACML Policies. IEEE Transactions on Knowledge and Data Engineering, 25(9), pp. 1946-1959.

9. Liu, A. X., Chen, F., Hwang, J., and Xie, T. 2011. Designing Fast and Scalable XACML Policy Evaluation Engines. IEEE Transactions on Computers, 60(12), pp. 1802-1817.

10. Ngo, C., Demchenko, Y., and Laat, C. D. 2015. Decision Diagrams for XACML Policy Evaluation and Management. Journal of Computers and Security, 49, pp. 1-16.

11. Priebe, T., Dobmeier, W., Schläger, C., and Kamprath, N. 2007. Supporting Attribute Based Access Control in Authorization and Authentication Infrastructures with Ontologies. Journal of Software, 2(1), pp. 27-38.

12. Shaikh, R. A., Adi, K., and Logrippo, L. 2016. A Data Classification Method for Inconsistency and Incompleteness Detection in Access Control Policy Sets. International Journal of Information Security, pp. 1-23.

13. Singh, K. and Singh, S. 2010. Design and Evaluation of XACML Conflict Policies Detection Mechanism International Journal of Computer Science and Information

Technology, 2, pp. 65-74.

14. Adi, K., Bouzida, Y., Hattak, I., Logrippo, L., and Mankovskii, S. 2009. Typing for Conflict Detection in Access Control Policies. Proceedings of the 4th International Conference on E-Technologies (MCETECH), pp. 212-226.

15. Bertino, E., Buccafurri, F., Ferrari, E., and Rullo, P. 1998. An Authorization Model and its Formal Semantics. Proceedings of the 5th European Symposium on Research in Computer Security (ESORICS), pp. 127-142.

16. Damiani, E., di Vimercati, S. D. C., Fugazza, C., and Samarati, P. 2006. Modality Conflicts in Semantics Aware Access Control. Proceedings of the 6th International Conference on Web Engineering (ICWE), pp. 249-256.

17. Dong, C., Russello, G., and Dulay, N. 2008. Flexible Resolution of Authorisation Conflicts in Distributed Systems. Proceedings of the 19th International Workshop on Distributed Systems: Operations and Management (DSOM), pp. 95-108.

18. Fatema, K. and Chadwick, D. 2014. Resolving Policy Conflicts-Integrating Policies from Multiple Authors. Proceedings of the International Conference on Advanced Information Systems Engineering (CAiSE), pp. 310-321.

19. Kamoda, H., Yamaoka, M., Matsuda, S., Broda, K., and Sloman, M. 2005. Policy Conflict Analysis using Free Variable Tableaux for Access Control in WebServices Environments. Proceedings of the 14th International World Wide Web Conference (WWW), pp. 121-126.

20. Teo, P. K., Ibrahim, H., Udzir, N. I., and Sidi, F. 2013. Heterogeneity XACML Policy Evaluation Engine. Proceedings of the 2nd International Conference on Digital Enterprise and Information Systems(DEIS), pp. 230-238.

21. Mohan, A., Blough, D. M., Kurc, T., Post, A., and Saltz, J. 2011. Detection of Conflicts and Inconsistencies in Taxonomy Based Authorization Policies. Proceedings of the 2011 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 590-594.

22. Neri, M. A., Guarnieri, M., Magri, E., Mutti, S., and Paraboschi, S. 2012. Conflict Detection in Security Policies using Semantic Web Technology. Proceedings of the 1st AESS European Conference on Satellite Telecommunications (ESTEL), pp. 1-6.

23. Reul, Q. and Zhao, G. 2010. Enabling Access to Web Resources through SecPODE-based Annotations. Proceedings of the 2010 Confederated International Conferences on the Move to Meaningful Internet Systems (OTM), pp. 596-605.

24. Russello, G., Dong, C., and Dulay, N. 2007. Authorisation and Conflict Resolution for Hierarchical Domains. Proceedings of the Eighth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY), pp. 201-210.

25. Stepien, B. and Felty, A. 2016. Using Expert Systems to Statically Detect "Dynamic" Conflicts in XACML. Proceedings of the 11th International Conference on Availability, Reliability and Security (ARES).

26. Xia, X. 2012. A Conflict Detection Approach for XACML Policies on Hierarchical Resources. Proceedings of the 2012 IEEE International Conference on Green Computing and Communications (GREENCOM), pp. 755-760.

27. Proctor, S. 2004. Sun's XACML implementation. http://sunxacml.sourceforge.net/.